

(19) 世界知的所有権機関
国際事務局(43) 国際公開日
2001年11月1日 (01.11.2001)

PCT

(10) 国際公開番号
WO 01/82604 A1

- (51) 国際特許分類: H04N 5/91 (72) 発明者; および
(75) 発明者/出願人 (米国についてのみ): 加藤元樹 (KATO, Motoki) [JP/JP]. 浜田俊也 (HAMADA, Toshiya) [JP/JP]; 〒141-0001 東京都品川区北品川6丁目7番35号 ソニー株式会社内 Tokyo (JP).
- (21) 国際出願番号: PCT/JP01/03413
- (22) 国際出願日: 2001年4月20日 (20.04.2001)
- (25) 国際出願の言語: 日本語 (74) 代理人: 小池 晃, 外 (KOIKE, Akira et al.); 〒105-0001 東京都港区虎ノ門二丁目6番4号 第11森ビル Tokyo (JP).
- (26) 国際公開の言語: 日本語 (81) 指定国 (国内): CN, KR, US.
- (30) 優先権データ:
特願2000-183770 2000年4月21日 (21.04.2000) JP 添付公開書類:
特願2000-268042 2000年9月5日 (05.09.2000) JP -- 国際調査報告書
- (71) 出願人 (米国を除く全ての指定国について): ソニー株式会社 (SONY CORPORATION) [JP/JP]; 〒141-0001 東京都品川区北品川6丁目7番35号 Tokyo (JP). 2文字コード及び他の略語については、定期発行される各PCTガゼットの巻頭に掲載されている「コードと略語のガイダンスノート」を参照。

(54) Title: INFORMATION PROCESSING APPARATUS AND METHOD, PROGRAM, AND RECORDED MEDIUM

(54) 発明の名称: 情報処理装置及び方法、プログラム、並びに記録媒体

シンタックス	バイト数	略号
ClipInfo01		
version_number	8*4	bslbf
length	32	uimabf
Clip_stream_type	8	bslbf
offset_SPN	32	uimabf
TS_recording_rule	24	uimabf
reserved	8	bslbf
record_time_and_date	4*14	bslbf
reserved	8	bslbf
duration	4*5	bslbf
reserved	7	bslbf
time_controlled_flag	1	bslbf
TS_average_rate	24	uimabf
if (Clip_stream_type == 1) // Bridge-Clip AV stream		
RSPN_arrival_time_discontinuity	32	uimabf
else		
reserved	32	bslbf
reserved_for_system_use	144	bslbf
reserved	11	bslbf
is_format_identifier_valid	1	bslbf
is_original_network_ID_valid	1	bslbf
is_transport_stream_ID_valid	1	bslbf
is_service_ID_valid	1	bslbf
is_country_code_valid	1	bslbf
format_identifier	32	bslbf
original_network_ID	16	uimabf
transport_stream_ID	16	uimabf
service_ID	16	uimabf
country_code	24	bslbf
stream_format_name	16*8	bslbf
reserved_for_future_use	256	bslbf

a...SYNTAX

b...NUMBER OF BYTES

c...SYMBOL

(57) Abstract: An AV stream, which is an entity of content, is managed by ClipInformation, and reproduction of the AV stream is managed by PlayList. Address information RSPN_arrival_time_discontinuity in a discontinuous point in the AV stream, which is attribute information on the AV stream, information EP_map, TU_map for relating the time information in the AV stream to the address information, and time information ClipMark on a characteristic image in the AV stream are recorded in the ClipInformation, thereby enabling quick search for a predetermined mark.

WO 01/82604 A1

[続葉有]



(57) 要約:

コンテンツの実体としてのAVストリームは、ClipInformationにより管理され、AVストリームの再生は、PlayListにより管理される。AVストリームの属性情報としての、AVストリーム中の不連続点のアドレス情報RSPN_arrival_time_discontinuity、AVストリーム中の時刻情報とアドレス情報を関連づける情報EP_map, TU_map、並びに、AVストリーム中の特徴的な画像の時刻情報ClipMarkは、ClipInformationに記録される。これにより、所定のマークを迅速に検索できるようにする。

明細書

情報処理装置及び方法、プログラム、並びに記録媒体

技術分野

本発明は情報処理装置及び方法、並びにプログラムに関し、特に、AVストリーム内のIピクチャのアドレス情報、符号化パラメータ、変化点情報、マークなどの情報をファイルとして記録する情報処理装置及び方法、記録媒体、並びにプログラムに関する。

背景技術

近年、記録再生装置から取り外し可能なディスク型の記録媒体として、各種の光ディスクが提案されつつある。このような記録可能な光ディスクは、数ギガバイトの大容量メディアとして提案されており、ビデオ信号等のAV (Audio Visual) 信号を記録するメディアとしての期待が高い。この記録可能な光ディスクに記録するデジタルのAV信号のソース（供給源）としては、CSデジタル衛星放送やBSデジタル放送があり、また、将来はデジタル方式の地上波テレビジョン放送等も提案されている。

ここで、これらのソースから供給されるデジタルビデオ信号は、通常MPEG (Moving Picture Experts Group) 2方式で画像圧縮されているのが一般的である。また、記録装置には、その装置固有の記録レートが定められている。従来の民生用映像蓄積メディアで、デジタル放送のデジタルビデオ信号を記録する場合、アナログ記録方式であれば、デジタルビデオ信号をデコード後、帯域制限をして記録する。あるいは、MPEG1 Video、MPEG2 Video、DV方式をはじめとするデジタル記録方式であれば、1度デコードされた後に、その装置固有の記録レート符号化方式で再エンコードされて記録される。

しかしながら、このような記録方法は、供給されたビットストリームを1度デ

コードし、その後で帯域制限や再エンコードを行って記録するため、画質の劣化を伴う。画像圧縮されたデジタル信号の記録をする場合、入力されたデジタル信号の伝送レートが記録再生装置の記録レートを超えない場合には、供給されたビットストリームをデコードや再エンコードすることなく、そのまま記録する方法が最も画質の劣化が少ない。ただし、画像圧縮されたデジタル信号の伝送レートが記録媒体としてのディスクの記録レートを超える場合には、記録再生装置でデコード後、伝送レートがディスクの記録レートの上限以下になるように、再エンコードをして記録する必要がある。

また、入力デジタル信号のビットレートが時間により増減する可変レート方式によって伝送されている場合には、回転ヘッドが固定回転数であるために記録レートが固定レートになるテープ記録方式に比べ、1度バッファにデータを蓄積し、バースト的に記録ができるディスク記録装置の方が記録媒体の容量をより無駄なく利用できる。

以上のように、デジタル放送が主流となる将来においては、データストリーマのように放送信号をデジタル信号のまま、デコードや再エンコードすることなく記録し、記録媒体としてディスクを使用した記録再生装置が求められると予測される。

上述したような装置により、複数のデータ（例えば、映像データや音声データなどから構成される番組のデータ）が記録されている記録媒体を再生する際、ユーザのランダムアクセスや特殊再生の指示に対して、記録媒体からのAVストリームの読出位置の決定やストリームの復号といった処理を速やかに行わなくてはならないが、記録媒体に記録されるデータ量が増加するに従い、そのような処理を速やかにできないといった課題があった。

発明の開示

本発明はこのような状況に鑑みてなされたものであり、AVストリーム内のIピクチャのアドレス情報、符号化パラメータ、変化点情報、マークなどの情報をファイルとして記録することにより、AVストリームの読出位置の決定や復号処理を

速やかに行えるようにし、特に、所定のマークを迅速に検索できるようにすることを目的とする。

本発明に係る情報処理装置は、AVストリーム中の符号化情報が連続な区間の開始アドレス情報、AVストリーム中の時刻情報とアドレス情報を関連付ける情報、及びAVストリーム中の特徴的な画像の時刻情報を生成する生成手段と、生成手段により生成された情報をClip情報として記録媒体に記録する記録手段とを備える。

符号化情報が連続な区間の開始アドレス情報は、STCシーケンス又はプログラムシーケンスの開始アドレスであり、時刻情報とアドレス情報を関連付ける情報は、EP_map又はTU_mapであり、特徴的な画像の時刻情報は、ClipMarkとすることができる。

記録手段は、AVストリームの記録レートの平均値に関する情報を記録媒体にさらに記録することができる。

平均値に関する情報は、TS_average_rateとすることができる。

AVストリームはトランスポートストリームとすることができる。

AVストリーム中の符号化情報が連続な区間の開始アドレス情報は、トランスポートストリームのシステムタイムクロックが連続な区間であるSTCシーケンスの開始アドレスを含むようにすることができる。

AVストリーム中の符号化情報が連続な区間の開始アドレス情報は、トランスポートストリームのプログラム内容が一定な区間であるプログラムシーケンスの開始アドレスを含むようにすることができる。

AVストリーム中の符号化情報が連続な区間の開始アドレス情報は、トランスポートストリームのトランスポートパケットの到着時間に基づくアライバルタイムが連続な区間の開始アドレスを含むようにすることができる。

AVストリーム中の時刻情報とアドレス情報を関連付ける情報は、トランスポートストリームのIピクチャのアドレスとそのプレゼンテーションタイムスタンプを含むようにすることができる。

AVストリーム中の時刻情報とアドレス情報を関連付ける情報の少なくとも一部を圧縮する圧縮手段をさらに備え、記録手段は、圧縮手段により圧縮された情報を記録することができる。

AVストリーム中の時刻情報とアドレス情報を関連付ける情報は、トランスポートパケットの到着時刻に基づいたアライバルタイムと、それに対応するトランスポートパケットのAVストリームデータ中のアドレスを含むようにすることができる。

本発明に係る情報処理方法は、AVストリーム中の符号化情報が連続な区間の開始アドレス情報、AVストリーム中の時刻情報とアドレス情報を関連付ける情報、及びAVストリーム中の特徴的な画像の時刻情報を生成する生成ステップと、生成ステップにより生成された情報をClip情報として記録媒体に記録する記録ステップとを含む。

本発明に係る記録媒体のプログラムは、AVストリーム中の符号化情報が連続な区間の開始アドレス情報、AVストリーム中の時刻情報とアドレス情報を関連付ける情報、及びAVストリーム中の特徴的な画像の時刻情報を生成する生成ステップと、生成ステップにより生成された情報をClip情報として記録媒体に記録する記録ステップとを含む。

本発明に係るプログラムは、AVストリーム中の符号化情報が連続な区間の開始アドレス情報、AVストリーム中の時刻情報とアドレス情報を関連付ける情報、及びAVストリーム中の特徴的な画像の時刻情報を生成する生成ステップと、生成ステップにより生成された情報をClip情報として記録媒体に記録する記録ステップとを実行させる。

本発明に係る情報処理装置は、Clip情報として、AVストリーム中の符号化情報が連続な区間の開始アドレス情報、AVストリーム中の時刻情報とアドレス情報を関連付ける情報、及びAVストリーム中の特徴的な画像の時刻情報を再生する再生手段と、再生手段により再生されたClip情報に基づいて、AVストリームの出力を制御する制御手段とを備える。

本発明に係る情報処理方法は、Clip情報として、AVストリーム中の符号化情報が連続な区間の開始アドレス情報、AVストリーム中の時刻情報とアドレス情報を関連付ける情報、及びAVストリーム中の特徴的な画像の時刻情報を再生する再生ステップと、再生ステップの処理により再生されたClip情報に基づいて、AVストリームの出力を制御する制御ステップとを含む。

本発明に係る記録媒体のプログラムは、Clip情報として、AVストリーム中の符号化情報が連続な区間の開始アドレス情報、AVストリーム中の時刻情報とアドレス情報を関連付ける情報、及びAVストリーム中の特徴的な画像の時刻情報を再生する再生ステップと、再生ステップの処理により再生されたClip情報に基づいて、AVストリームの出力を制御する制御ステップとを含む。

本発明に係るプログラムは、Clip情報として、AVストリーム中の符号化情報が連続な区間の開始アドレス情報、AVストリーム中の時刻情報とアドレス情報を関連付ける情報、及びAVストリーム中の特徴的な画像の時刻情報を再生する再生ステップと、再生ステップの処理により再生されたClip情報に基づいて、AVストリームの出力を制御する制御ステップとを実行させる。

本発明に係る記録媒体は、Clip情報として、AVストリーム中の符号化情報が連続な区間の開始アドレス情報、AVストリーム中の時刻情報とアドレス情報を関連付ける情報、及びAVストリーム中の特徴的な画像の時刻情報が記録されている。

本発明では、Clip情報として、AVストリーム中の符号化情報が連続な区間の開始アドレス情報、AVストリーム中の時刻情報とアドレス情報を関連付ける情報、及びAVストリーム中の特徴的な画像の時刻情報が記録される。

また、本発明では、Clip情報として、AVストリーム中の符号化情報が連続な区間の開始アドレス情報、AVストリーム中の時刻情報とアドレス情報を関連付ける情報、及びAVストリーム中の特徴的な画像の時刻情報が再生される。

図面の簡単な説明

図1は、本発明を適用した記録再生装置の一実施の形態の構成を示す図である。

図2は、記録再生装置1により記録媒体に記録されるデータのフォーマットについて説明する図である。

図3は、Real PlayListとVirtual PlayListについて説明する図である。

図4A、4B、4Cは、Real PlayListの作成について説明する図である。

図5A、5B、5Cは、Real PlayListの削除について説明する図である。

図 6 A, 6 B は、アセンブル編集について説明する図である。

図 7 は、Virtual PlayList にサブバスを設ける場合について説明する図である。

図 8 は、PlayList の再生順序の変更について説明する図である。

図 9 は、PlayList 上のマークと Clip 上のマークについて説明する図である。

図 10 は、メニューサムネイルについて説明する図である。

図 11 は、PlayList に付加されるマークについて説明する図である。

図 12 は、クリップに付加されるマークについて説明する図である。

図 13 は、PlayList、Clip、サムネイルファイルの関係について説明する図である。

図 14 は、ディレクトリ構造について説明する図である。

図 15 は、info.dvr のシンタクスを示す図である。

図 16 は、DVRVolume のシンタクスを示す図である。

図 17 は、ResumeVolume のシンタクスを示す図である。

図 18 は、UIAppInfoVolume のシンタクスを示す図である。

図 19 は、CharactersetValue のテーブルを示す図である。

図 20 は、TableOfPlayList のシンタクスを示す図である。

図 21 は、TableOfPlayList の他のシンタクスを示す図である。

図 22 は、MakersPrivateData のシンタクスを示す図である。

図 23 は、xxxxx.rpls と yyyyy.vpls のシンタクスを示す図である。

図 24 A, 24 B, 24 C は、PlayList について説明する図である。

図 25 は、PlayList のシンタクスを示す図である。

図 26 は、PlayList_type のテーブルを示す図である。

図 27 は、UIAppInfoPlayList のシンタクスを示す図である。

図 28 A, 28 B, 28 C は、図 27 に示した UIAppInfoPlayList のシンタクス内のフラグについて説明する図である。

図 29 は、PlayItem について説明する図である。

図 30 は、PlayItem について説明する図である。

図 31 は、PlayItem について説明する図である。

図 3 2 は、PlayItemのシンタクスを示す図である。

図 3 3 は、IN_timeについて説明する図である。

図 3 4 は、OUT_timeについて説明する図である。

図 3 5 は、Connection_Conditionのテーブルを示す図である。

図 3 6 A, 図 3 6 B, 図 3 6 C, 図 3 6 D は、Connection_Conditionについて説明する図である。

図 3 7 は、BridgeSequenceInfoを説明する図である。

図 3 8 は、BridgeSequenceInfoのシンタクスを示す図である。

図 3 9 は、SubPlayItemについて説明する図である。

図 4 0 は、SubPlayItemのシンタクスを示す図である。

図 4 1 は、SubPath_typeのテーブルを示す図である。

図 4 2 は、PlayListMarkのシンタクスを示す図である。

図 4 3 は、Mark_typeのテーブルを示す図である。

図 4 4 は、Mark_time_stampを説明する図である。

図 4 5 は、zzzzz.clipのシンタクスを示す図である。

図 4 6 は、ClipInfoのシンタクスを示す図である。

図 4 7 は、Clip_stream_typeのテーブルを示す図である。

図 4 8 は、offset_SPNについて説明する図である。

図 4 9 は、offset_SPNについて説明する図である。

図 5 0 A, 図 5 0 B は、STC区間について説明する図である。

図 5 1 は、STC_Infoについて説明する図である。

図 5 2 は、STC_Infoのシンタクスを示す図である。

図 5 3 は、ProgramInfoを説明する図である。

図 5 4 は、ProgramInfoのシンタクスを示す図である。

図 5 5 は、VideoCondngInfoのシンタクスを示す図である。

図 5 6 は、Video_formatのテーブルを示す図である。

図 5 7 は、frame_rateのテーブルを示す図である。

図 5 8 は、display_aspect_ratioのテーブルを示す図である。

図 5 9 は、AudioCondngInfoのシンタクスを示す図である。

- 図 6 0 は、audio_codingのテーブルを示す図である。
- 図 6 1 は、audio_component_typeのテーブルを示す図である。
- 図 6 2 は、sampling_frequencyのテーブルを示す図である。
- 図 6 3 は、CPIについて説明する図である。
- 図 6 4 は、CPIについて説明する図である。
- 図 6 5 は、CPIのシンタクスを示す図である。
- 図 6 6 は、CPI_typeのテーブルを示す図である。
- 図 6 7 は、ビデオEP_mapについて説明する図である。
- 図 6 8 は、EP_mapについて説明する図である。
- 図 6 9 は、EP_mapについて説明する図である。
- 図 7 0 は、EP_mapのシンタクスを示す図である。
- 図 7 1 は、EP_typevaluesのテーブルを示す図である。
- 図 7 2 は、EP_map_for_one_stream_PIDのシンタクスを示す図である。
- 図 7 3 は、TU_mapについて説明する図である。
- 図 7 4 は、TU_mapのシンタクスを示す図である。
- 図 7 5 は、ClipMarkのシンタクスを示す図である。
- 図 7 6 は、Mark_typeのテーブルを示す図である。
- 図 7 7 は、Mark_type_stampのテーブルを示す図である。
- 図 7 8 は、ClipMarkのシンタクスの他の例を示す図である。
- 図 7 9 は、Mark_typeのテーブルの他の例を示す図である。
- 図 8 0 は、mark_entry()とrepresentative_picture_entry()の例を示す図である。
- 図 8 1 は、mark_entry()とrepresentative_picture_entry()のシンタクスを示す図である。
- 図 8 2 は、mark_entry()とrepresentative_picture_entry()のシンタクスの他の例を示す図である。
- 図 8 3 は、RSPN_ref_EP_startとoffset_num_picturesの関係を説明する図である。
- 図 8 4 は、mark_entry()とrepresentative_picture_entry()のシンタクスの他

の例を示す図である。

図 8 5 は、ClipMarkとEP_mapの関係を説明する図である。

図 8 6 は、menu.thmbとmark.thmbのシンタクスを示す図である。

図 8 7 は、Thumbnailのシンタクスを示す図である。

図 8 8 は、thumbnail_picture_formatのテーブルを示す図である。

図 8 9 A, 図 8 9 B は、tn_blockについて説明する図である。

図 9 0 は、DVRMPEG2のトランスポートストリームの構造について説明する図である。

図 9 1 は、DVRMPEG2のトランスポートストリームのレコーダモデルを示す図である。

図 9 2 は、DVRMPEG2のトランスポートストリームのプレーヤモデルを示す図である。

図 9 3 は、sourcepacketのシンタクスを示す図である。

図 9 4 は、TP_extra_headerのシンタクスを示す図である。

図 9 5 は、copypermissionindicatorのテーブルを示す図である。

図 9 6 は、シームレス接続について説明する図である。

図 9 7 は、シームレス接続について説明する図である。

図 9 8 は、シームレス接続について説明する図である。

図 9 9 は、シームレス接続について説明する。

図 1 0 0 は、シームレス接続について説明する図である。

図 1 0 1 は、オーディオのオーバーラップについて説明する図である。

図 1 0 2 は、BridgeSequenceを用いたシームレス接続について説明する図である。

図 1 0 3 は、BridgeSequenceを用いないシームレス接続について説明する図である。

図 1 0 4 は、DVRSTDモデルを示す図である。

図 1 0 5 は、復号、表示のタイミングチャートを示す図である。

図 1 0 6 は、図 8 1 のシンタクスの場合におけるマーク点で示されるシーンの頭出し再生を説明するフローチャートである。

図 1 0 7 は、図 7 5 又は図 7 8 のClipMarkのmark_entry()/representative_picture_entry()が図 8 1 のシンタクスの場合における再生の動作を説明する図である。

図 1 0 8 は、EP_mapの例を示す図である。

図 1 0 9 は、ClipMarkの例を示す図である。

図 1 1 0 は、図 7 5 又は図 7 8 のClipMarkのmark_entry()/representative_picture_entry()が図 8 1 のシンタクスの場合におけるCMスキップ再生処理を説明するフローチャートである。

図 1 1 1 は、図 7 5 又は図 7 8 のClipMarkのmark_entry()/representative_picture_entry()が図 8 1 のシンタクスの場合におけるCMスキップ再生処理を説明するフローチャートである。

図 1 1 2 は、図 7 5 又は図 7 8 のClipMarkのmark_entry()/representative_picture_entry()が図 8 2 のシンタクスの場合におけるマーク点で示されるシーンの頭出し再生を説明するフローチャートである。

図 1 1 3 は、図 7 5 又は図 7 8 のClipMarkのmark_entry()/representative_picture_entry()が図 8 2 のシンタクスの場合における再生を説明する図である。

図 1 1 4 は、EP_mapの例を示す図である。

図 1 1 5 は、ClipMarkの例を示す図である。

図 1 1 6 は、図 7 5 又は図 7 8 のClipMarkのmark_entry()/representative_picture_entry()が図 8 2 のシンタクスの場合におけるCMスキップ再生を説明するフローチャートである。

図 1 1 7 は、図 7 5 又は図 7 8 のClipMarkのmark_entry()/representative_picture_entry()が図 8 2 のシンタクスの場合におけるCMスキップ再生を説明するフローチャートである。

図 1 1 8 は、図 7 5 又は図 7 8 のClipMarkのmark_entry()/representative_picture_entry()が図 8 4 のシンタクスの場合におけるマーク点で示されるシーンの頭出し再生を説明するフローチャートである。

図 1 1 9 は、図 7 5 又は図 7 8 のClipMarkのmark_entry()/representative_picture_entry()が図 8 4 のシンタクスの場合における再生を説明する図である。

図 1 2 0 は、EP_mapの例を示す図である。

図 1 2 1 は、ClipMarkの例を示す図である。

図 1 2 2 は、図 7 5 又は図 7 8 のClipMarkのmark_entry()/representative_picture_entry()が図 8 4 のシンタクスの場合におけるCMスキップ再生を説明するフローチャートである。

図 1 2 3 は、図 7 5 又は図 7 8 のClipMarkのmark_entry()/representative_picture_entry()が図 8 4 のシンタクスの場合におけるCMスキップ再生を説明するフローチャートである。

図 1 2 4 は、アプリケーションフォーマットを示す図である。

図 1 2 5 は、PlayList上のマークとClip上のマークを説明する図である。

図 1 2 6 は、ClipMarkのシンタクスの他の例を示す図である。

図 1 2 7 は、ClipMarkのシンタクスのさらに他の例を示す図である。

図 1 2 8 は、ClipInfo()のシンタクスの別の例を示す図である。

図 1 2 9 は、ProgramInfo()のシンタクスの別の例を示す図である。

図 1 3 0 は、StreamCodingInfo()のシンタクスを示す図である。

図 1 3 1 は、stream_coding_typeを説明する図である。

図 1 3 2 は、EP-fineとEP-coarseの関係を説明する図である。

図 1 3 3 は、PTS_EP_fineとPTS_EP_coarseのフォーマットを説明する図である。

図 1 3 4 は、RSPN_EP_fineとRSPN_EP_coarseのフォーマットを説明する図である。

図 1 3 5 は、EP-coarseのエントリとEP-fineのエントリを説明する図である。

図 1 3 6 は、EP_mapのシンタクスの別の例を示す図である。

図 1 3 7 は、EP_stream_typevaluesを説明する図である。

図 1 3 8 は、図 1 3 6 のEP_mapのEP_map_for_one_stream_PIDのシンタクスを示す図である。

図 1 3 9 は、EP_video_typeの値の意味を説明する図である。

図 1 4 0 は、ClipAVストリームファイル及びClipInformationファイルの作成処理を説明するフローチャートである。

図 1 4 1 は、STC_Infoの作成の動作例を説明するフローチャートである。

図 1 4 2 は、ProgramInfoの作成の動作例を説明するフローチャートである。

図 1 4 3 は、EP_mapの作成の動作例を説明するフローチャートである。

図 1 4 4 は、アナログAV信号をエンコードして記録する場合における、図 7 5 又は図 7 8 のClipMarkのmark_entry()/representative_picture_entry()が図 8 1 に示すシンタクスであるときのClipMarkの作成方法を説明するフローチャートである。

図 1 4 5 は、デジタルインタフェースから入力されたトランスポートストリームを記録する場合における、図 7 5 又は図 7 8 のClipMarkのmark_entry()/representative_picture_entry()が図 8 1 に示すシンタクスであるときのClipMarkの作成方法を説明するフローチャートである。

図 1 4 6 は、EP_mapを使う特殊再生を説明する図である。

図 1 4 7 は、EP_mapを使用した I ピクチャサーチのためのプレーヤモデルを説明する図である。

図 1 4 8 は、ミニマイズのオペレーションの例を示す図である。

図 1 4 9 は、ミニマイズのときにIN_timeの前の不要なストリームデータを消去する例を示す図である。

図 1 5 0 は、ミニマイズのときにOUT_timeの後ろの不要なストリームデータを消去する例を説明する図である。

図 1 5 1 は、媒体を説明する図である。

発明を実施するための最良の形態

以下に、本発明の実施の形態について、図面を参照して説明する。図 1 は、本発明を適用した記録再生装置 1 の内部構成例を示す図である。まず、外部から入力された信号を記録媒体に記録する動作を行う記録部 2 の構成について説明する。記録再生装置 1 は、アナログデータ又はデジタルデータを入力し、記録することができる構成とされている。

端子 1 1 には、アナログのビデオ信号が、端子 1 2 には、アナログのオーディオ信号が、それぞれ入力される。端子 1 1 に入力されたビデオ信号は、解析部 1

4とAVエンコーダ15に、それぞれ出力される。端子12に入力されたオーディオ信号は、解析部14とAVエンコーダ15に出力される。解析部14は、入力されたビデオ信号とオーディオ信号からシーンチェンジなどの特徴点を抽出する。

AVエンコーダ15は、入力されたビデオ信号とオーディオ信号を、それぞれ符号化し、符号化ビデオストリーム(V)、符号化オーディオストリーム(A)、及びAV同期等のシステム情報(S)をマルチプレクサ16に出力する。

符号化ビデオストリームは、例えば、MPEG (Moving Picture Expert Group) 2方式により符号化されたビデオストリームであり、符号化オーディオストリームは、例えば、MPEG1方式により符号化されたオーディオストリームや、ドルビーAC3方式(商標)により符号化されたオーディオストリーム等である。マルチプレクサ16は、入力されたビデオ及びオーディオのストリームを、入力システム情報に基づいて多重化して、スイッチ17を介して多重化ストリーム解析部18とソースパケッタイザ19に出力する。

多重化ストリームは、例えば、MPEG-2トランスポートストリームやMPEG2プログラムストリームである。ソースパケッタイザ19は、入力された多重化ストリームを、そのストリームを記録させる記録媒体100のアプリケーションフォーマットに従って、ソースパケットから構成されるAVストリームに符号化する。AVストリームは、ECC(誤り訂正)符号化部20と変調部21でECC符号の付加と変調処理が施され、書込部22に出力される。書込部22は、制御部23から出力される制御信号に基づいて、記録媒体100にAVストリームファイルを書き込む(記録する)。

デジタルインタフェース又はデジタルテレビジョンチューナから入力されるデジタルテレビジョン放送等のトランスポートストリームは、端子13に入力される。端子13に入力されたトランスポートストリームの記録方式には、2通りあり、それらは、トランスペアレントに記録する方式と、記録ビットレートを下げるなどの目的のために再エンコードをした後に記録する方式である。記録方式の指示情報は、ユーザインタフェースとしての端子24から制御部23へ入力される。

入力トランスポートストリームをトランスペアレントに記録する場合、端子1

3に入力されたトランスポートストリームは、スイッチ17を介して多重化ストリーム解析部18と、ソースパケットタイザ19に出力される。これ以降の記録媒体100へAVストリームが記録されるまでの処理は、上述のアナログの入力オーディオ信号とビデオ信号を符号化して記録する場合と同一の処理なので、その説明は省略する。

入力トランスポートストリームを再エンコードした後に記録する場合、端子13に入力されたトランスポートストリームは、デマルチプレクサ26に入力される。デマルチプレクサ26は、入力されたトランスポートストリームに対してデマルチプレクス処理を施し、ビデオストリーム(V)、オーディオストリーム(A)及びシステム情報(S)を抽出する。

デマルチプレクサ26により抽出されたストリーム(情報)のうち、ビデオストリームはAVデコーダ27に、オーディオストリームとシステム情報はマルチプレクサ16に、それぞれ出力される。AVデコーダ27は、入力されたビデオストリームを復号し、その再生ビデオ信号をAVエンコーダ15に出力する。AVエンコーダ15は、入力ビデオ信号を符号化し、符号化ビデオストリーム(V)をマルチプレクサ16に出力する。

一方、デマルチプレクサ26から出力され、マルチプレクサ16に入力されたオーディオストリームとシステム情報、及びAVエンコーダ15から出力されたビデオストリームは、入力システム情報に基づいて、多重化されて、多重化ストリームとして多重化ストリーム解析部18とソースパケットタイザ19にスイッチ17を介して出力される。これ以後の記録媒体100へAVストリームが記録されるまでの処理は、上述のアナログの入力オーディオ信号とビデオ信号を符号化して記録する場合と同一の処理なので、その説明は省略する。

本実施の形態の記録再生装置1は、AVストリームのファイルを記録媒体100に記録するとともに、そのファイルを説明するアプリケーションデータベース情報も記録する。アプリケーションデータベース情報は、制御部23により作成される。制御部23への入力情報は、解析部14からの動画像の特徴情報、多重化ストリーム解析部18からのAVストリームの特徴情報、及び端子24から入力されるユーザからの指示情報である。

解析部 1 4 から供給される動画像の特徴情報は、AVエンコーダ 1 5 がビデオ信号を符号化する場合において、解析部 1 4 により生成されるものである。解析部 1 4 は、入力ビデオ信号とオーディオ信号の内容を解析し、入力動画像信号の中の特徴的な画像（クリップマーク）に関係する情報を生成する。これは、例えば、入力ビデオ信号の中のプログラムの開始点、シーンチェンジ点やCMコマercialのスタート点、エンド点、タイトルやテロップなどの特徴的なクリップマーク点の画像の指示情報であり、また、それにはその画像のサムネイルも含まれる。さらにオーディオ信号のステレオとモノラルの切換点や、無音区間などの情報も含まれる。

これらの画像の指示情報は、制御部 2 3 を介して、マルチプレクサ 1 6 へ入力される。マルチプレクサ 1 6 は、制御部 2 3 からクリップマークとして指定される符号化ピクチャを多重化するとき、その符号化ピクチャをAVストリーム上で特定するための情報を制御部 2 3 に返す。具体的には、この情報は、ピクチャのPTS（プレゼンテーションタイムスタンプ）又はその符号化ピクチャのAVストリーム上でのアドレス情報である。制御部 2 3 は、特徴的な画像の種類とその符号化ピクチャをAVストリーム上で特定するための情報を関連付けて記憶する。

多重化ストリーム解析部 1 8 からのAVストリームの特徴情報は、記録されるAVストリームの符号化情報に関係する情報であり、解析部 1 8 により生成される。例えば、AVストリーム内のIピクチャのタイムスタンプとアドレス情報、システムタイムクロックの不連続点情報、AVストリームの符号化パラメータ、AVストリームの中の符号化パラメータの変化点情報などが含まれる。また、端子 1 3 から入力されるトランスポートストリームをトランスペアレントに記録する場合、多重化ストリーム解析部 1 8 は、入力トランスポートストリームの中から前出のクリップマークの画像を検出し、その種類とクリップマークで指定するピクチャを特定するための情報を生成する。

端子 2 4 からのユーザの指示情報は、AVストリームの中の、ユーザが指定した再生区間の指定情報、その再生区間の内容を説明するキャラクタ文字、ユーザが好みのシーンにセットするブックマークやリジューム点の情報などである。

制御部 2 3 は、上述の入力情報に基づいて、AVストリームのデータベース（C1

ip)、AVストリームの再生区間(PlayItem)をグループ化したもの(PlayList)のデータベース、記録媒体100の記録内容の管理情報(info.dvr)、及びサムネイル画像の情報を作成する。これらの情報から構成されるアプリケーションデータベース情報は、AVストリームと同様にして、ECC符号化部20、変調部21で処理されて、書込部22へ入力される。書込部22は、制御部23から出力される制御信号に基づいて、記録媒体100へデータベースファイルを記録する。

上述したアプリケーションデータベース情報についての詳細は後述する。

このようにして記録媒体100に記録されたAVストリームファイル(画像データと音声データのファイル)と、アプリケーションデータベース情報が再生部3により再生される場合、まず、制御部23は、読出部28に対して、記録媒体100からアプリケーションデータベース情報を読み出すように指示する。そして、読出部28は、記録媒体100からアプリケーションデータベース情報を読み出し、そのアプリケーションデータベース情報は、復調部29とECC復号部30の復調と誤り訂正処理を経て、制御部23へ入力される。

制御部23は、アプリケーションデータベース情報に基づいて、記録媒体100に記録されているPlayListの一覧を端子24のユーザインタフェースへ出力する。ユーザは、PlayListの一覧から再生したいPlayListを選択し、再生を指定されたPlayListに関する情報が制御部23へ入力される。制御部23は、そのPlayListの再生に必要なAVストリームファイルの読出を、読出部28に指示する。読出部28は、その指示に従い、記録媒体100から対応するAVストリームを読み出し復調部29に出力する。復調部29に入力されたAVストリームは、所定の処理が施されることにより復調され、さらにECC復号部30の処理を経て、ソースデパケッタ31出力される。

ソースデパケッタ31は、記録媒体100から読み出され、所定の処理が施されたアプリケーションフォーマットのAVストリームを、デマルチプレクサ26が処理可能なストリームに変換する。デマルチプレクサ26は、制御部23により指定されたAVストリームの再生区間(PlayItem)を構成するビデオストリーム(V)、オーディオストリーム(A)及びAV同期等のシステム情報(S)を、AVデコーダ27に出力する。AVデコーダ27は、ビデオストリームとオーディオストリーム

を復号し、再生ビデオ信号と再生オーディオ信号を、それぞれ対応する端子32と端子33から出力する。

また、ユーザインタフェースとしての端子24から、ランダムアクセス再生や特殊再生を指示する情報が入力された場合、制御部23は、AVストリームのデータベース (Clip) の内容に基づいて、記憶媒体100からのAVストリームの読出位置を決定し、そのAVストリームの読出を、読出部28に指示する。例えば、ユーザにより選択されたPlayListを、所定の時刻から再生する場合、制御部23は、指定された時刻に最も近いタイムスタンプを持つIピクチャからのデータを読み出すように読出部28に指示する。

また、Clip Informationの中のClipMarkにストアされている番組の頭出し点やシーンチェンジ点の中から、ユーザがあるクリップマークを選択したとき（例えば、この動作は、ClipMarkにストアされている番組の頭出し点やシーンチェンジ点のサムネイル画像リストをユーザインタフェースに表示して、ユーザが、その中からある画像を選択することにより行われる）、制御部23は、Clip Informationの内容に基づいて、記録媒体100からのAVストリームの読出位置を決定し、そのAVストリームの読出を読出部28へ指示する。すなわち、ユーザが選択した画像がストアされているAVストリーム上でのアドレスに最も近いアドレスにあるIピクチャからのデータを読み出すように読出部28へ指示する。読出部28は、指定されたアドレスからデータを読み出し、読み出されたデータは、復調部29、ECC復号部30、ソースデパケッタイザ31の処理を経て、デマルチプレクサ26へ入力され、AVデコーダ27で復号されて、マーク点のピクチャのアドレスで示されるAVデータが再生される。

また、ユーザによって高速再生 (Fast-forward playback) が指示された場合、制御部23は、AVストリームのデータベース (Clip) に基づいて、AVストリームの中のI-ピクチャデータを順次連続して読み出すように読出部28に指示する。

読出部28は、指定されたランダムアクセスポイントからAVストリームのデータを読み出し、読み出されたデータは、後段の各部の処理を経て再生される。

次に、ユーザが、記録媒体100に記録されているAVストリームの編集をする場合を説明する。ユーザが、記録媒体100に記録されているAVストリームの再

生区間を指定して新しい再生経路を作成したい場合、例えば、番組Aという歌番組から歌手Aの部分を再生し、その後続けて、番組Bという歌番組の歌手Aの部分を再生したいといった再生経路を作成したい場合、ユーザインタフェースとしての端子24から再生区間の開始点（イン点）と終了点（アウト点）の情報が制御部23に入力される。制御部23は、AVストリームの再生区間（PlayItem）をグループ化したもの（PlayList）のデータベースを作成する。

ユーザが、記録媒体100に記録されているAVストリームの一部を消去したい場合、ユーザインタフェースとしての端子24から消去区間のイン点とアウト点の情報が制御部23に入力される。制御部23は、必要なAVストリーム部分だけを参照するようにPlayListのデータベースを変更する。また、AVストリームの不必要なストリーム部分を消去するように、書込部22に指示する。

ユーザが、記録媒体100に記録されているAVストリームの再生区間を指定して新しい再生経路を作成したい場合であり、かつ、それぞれの再生区間をシームレスに接続したい場合について説明する。このような場合、制御部23は、AVストリームの再生区間（PlayItem）をグループ化したもの（PlayList）のデータベースを作成し、さらに、再生区間の接続点付近のビデオストリームの部分的な再エンコードと再多重化を行う。

まず、端子24から再生区間のイン点のピクチャの情報と、アウト点のピクチャの情報が制御部23へ入力される。制御部23は、読出部28にイン点側ピクチャとアウト点側のピクチャを再生するために必要なデータの読出を指示する。そして、読出部28は、記録媒体100からデータを読み出し、そのデータは、復調部29、ECC復号部30、ソースデパケッタ31を経て、デマルチプレクサ26に出力される。

制御部23は、デマルチプレクサ26に入力されたデータを解析して、ビデオストリームの再エンコード方法（picture_coding_typeの変更、再エンコードする符号化ビット量の割り当て）と、再多重化方式を決定し、その方式をAVエンコーダ15とマルチプレクサ16に供給する。

次に、デマルチプレクサ26は、入力されたストリームをビデオストリーム（V）、オーディオストリーム（A）及びシステム情報（S）に分離する。ビデオストリー

ムは、AVデコーダ 27 に入力されるデータとマルチプレクサ 16 に入力されるデータがある。前者のデータは、再エンコードするために必要なデータであり、これはAVデコーダ 27 で復号され、復号されたピクチャはAVエンコーダ 15 で再エンコードされて、ビデオストリームにされる。後者のデータは、再エンコードをしないで、オリジナルのストリームからコピーされるデータである。オーディオストリーム、システム情報については、直接、マルチプレクサ 16 に入力される。

マルチプレクサ 16 は、制御部 23 から入力された情報に基づいて、入力ストリームを多重化し、多重化ストリームを出力する。多重化ストリームは、ECC符号化部 20、変調部 21 で処理されて、書込部 22 に入力される。書込部 22 は、制御部 23 から供給される制御信号に基づいて、記録媒体 100 にAVストリームを記録する。

以下に、アプリケーションデータベース情報や、その情報に基づく再生、編集といった操作に関する説明をする。図 2 は、アプリケーションフォーマットの構造を説明する図である。アプリケーションフォーマットは、AVストリームの管理のためにPlayListとClipの 2 つのレイヤをもつ。Volume Informationは、ディスク内の全てのClipとPlayListの管理をする。ここでは、1 つのAVストリームとその付属情報のペアを 1 つのオブジェクトと考え、それをClipと称する。AVストリームファイルはClip AV stream fileと称し、その付属情報は、Clip Information fileと称する。

1 つのClip AV stream fileは、MPEG-2トランスポートストリームをアプリケーションフォーマットによって規定される構造に配置したデータをストアする。一般的に、ファイルは、バイト列として扱われるが、Clip AV stream fileのコンテンツは、時間軸上に展開され、Clipの中のエントリポイント（Iピクチャ）は、主に時間ベースで指定される。所定のClipへのアクセスポイントのタイムスタンプが与えられたとき、Clip Information fileは、Clip AV stream fileの中でデータの読出を開始すべきアドレス情報を見つけるために役立つ。

PlayListについて、図 3 を参照して説明する。PlayListは、Clipの中からユーザが見たい再生区間を選択し、それを簡単に編集することができるようにするために設けられている。1 つのPlayListは、Clipの中の再生区間の集まりである。

所定のClipの中の1つの再生区間は、PlayItemと呼ばれ、それは、時間軸上のイン点 (IN) とアウト点 (OUT) の対で表される。したがって、PlayListは、複数のPlayItemが集まることにより構成される。

PlayListには、2つのタイプがある。1つは、Real PlayListであり、もう1つは、Virtual PlayListである。Real PlayListは、それが参照しているClipのストリーム部分を共有している。すなわち、Real PlayListは、その参照しているClipのストリーム部分に相当するデータ容量をディスクの中で占め、Real PlayListが消去された場合、それが参照しているClipのストリーム部分もまたデータが消去される。

Virtual PlayListは、Clipのデータを共有していない。したがって、Virtual PlayListが変更又は消去されたとしても、Clipの内容には何も変化が生じない。

次に、Real PlayListの編集について説明する。図4 Aは、Real PlayListのクリエイト (create : 作成) に関する図であり、AVストリームが新しいClipとして記録される場合、そのClip全体を参照するReal PlayListが新たに作成される操作である。

図4 Bは、Real PlayListのディバイド (divide : 分割) に関する図であり、Real PlayListが所望な点で分けられて、2つのReal PlayListに分割される操作である。この分割という操作は、例えば、1つのPlayListにより管理される1つのクリップ内に、2つの番組が管理されているような場合に、ユーザが1つ1つの番組として登録 (記録) し直したいといったようなときに行われる。この操作により、Clipの内容が変更される (Clip自体が分割される) ことはない。

図4 Cは、Real PlayListのコンバイン (combine : 結合) に関する図であり、2つのReal PlayListを結合して、1つの新しいReal PlayListにする操作である。この結合という操作は、例えば、ユーザが2つの番組を1つの番組として登録し直したいといったようなときに行われる。この操作により、Clipが変更される (Clip自体が1つにされる) ことはない。

図5 Aは、Real PlayList全体のデリート (delete : 削除) に関する図であり、所定のReal PlayList全体を消去する操作がされた場合、削除されたReal PlayListが参照するClipの、対応するストリーム部分も削除される。

図 5 B は、Real PlayList の部分的な削除に関する図であり、Real PlayList の所望な部分が削除された場合、対応する PlayItem が、必要な Clip のストリーム部分だけを参照するように変更される。そして、Clip の対応するストリーム部分は削除される。

図 5 C は、Real PlayList のミニマイズ (Minimize: 最小化) に関する図であり、Real PlayList に対応する PlayItem を、Virtual PlayList に必要な Clip のストリーム部分だけを参照するようにする操作である。Virtual PlayList にとって不必要な Clip の、対応するストリーム部分は削除される。

上述したような操作により、Real PlayList が変更されて、その Real PlayList が参照する Clip のストリーム部分が削除された場合、その削除された Clip を使用している Virtual PlayList が存在し、その Virtual PlayList において、削除された Clip により問題が生じる可能性がある。

そのようなことが生じないように、ユーザに、削除という操作に対して、「その Real PlayList が参照している Clip のストリーム部分を参照している Virtual PlayList が存在し、もし、その Real PlayList が消去されると、その Virtual PlayList もまた消去されることになるが、それでも良いか？」といったメッセージなどを表示させることにより、確認 (警告) を促した後に、ユーザの指示により削除の処理を実行又はキャンセルする。あるいは Virtual PlayList を削除する代わりに、Real PlayList に対してミニマイズの操作が行われるようにする。

次に Virtual PlayList に対する操作について説明する。Virtual PlayList に対して操作が行われたとしても、Clip の内容が変更されることはない。図 6 A、図 6 B は、アセンブル (Assemble) 編集 (IN-OUT 編集) に関する図であり、ユーザが見たいと所望した再生区間の PlayItem を作り、Virtual PlayList を作成するといった操作である。PlayItem 間のシームレス接続が、アプリケーションフォーマットによりサポートされている (後述)。

図 6 A に示したように、2 つの Real PlayList 1, 2 と、それぞれの Real PlayList に対応する Clip 1, 2 が存在している場合に、ユーザが Real PlayList 1 内の所定の区間 (In 1 乃至 Out 1 までの区間: PlayItem 1) を再生区間として指示し、続けて再生する区間として、Real PlayList 2 内の所定の区間 (In 2 乃至 Out 2 ま

での区間：PlayItem 2) を再生区間として指示したとき、図 6 B に示すように、PlayItem 1 と PlayItem 2 から構成される 1 つの Virtual PlayList が作成される。

次に、Virtual PlayList の再編集 (Re-editing) について説明する。再編集には、Virtual PlayList 中のイン点やアウト点の変更、Virtual PlayList への新しい PlayItem の挿入 (insert) や追加 (append)、Virtual PlayList 中の PlayItem の削除などがある。また、Virtual PlayList そのものを削除することもできる。

図 7 は、Virtual PlayList へのオーディオのアフレコ (Audio dubbing (post recording)) に関する図であり、Virtual PlayList へのオーディオのアフレコをサブパスとして登録する操作のことである。このオーディオのアフレコは、アプリケーションフォーマットによりサポートされている。Virtual PlayList のメインパスの AV ストリームに、付加的なオーディオストリームが、サブパスとして付加される。

Real PlayList と Virtual PlayList で共通の操作として、図 8 に示すような PlayList の再生順序の変更 (Moving) がある。この操作は、ディスク (ボリューム) 中での PlayList の再生順序の変更であり、アプリケーションフォーマットにおいて定義される Table Of PlayList (図 20 などを参照して後述する) によってサポートされる。この操作により、Clip の内容が変更されるようなことはない。

次に、マーク (Mark) について説明する。マークは、図 9 に示されるように、Clip 及び PlayList 中のハイライトや特徴的な時間を指定するために設けられている。Clip に付加されるマークは、ClipMark (クリップマーク) と呼ばれる。ClipMark は、AV ストリームの内容に起因する特徴的なシーンを指定する、例えば番組の頭出し点やシーンチェンジ点などである。ClipMark は、図 1 の例えば解析部 14 によって生成される。PlayList を再生するとき、その PlayList が参照する Clip のマークを参照して、使用することができる。

PlayList に付加されるマークは、PlayListMark (プレイリストマーク) と呼ばれる。PlayListMark は、主にユーザによってセットされる、例えば、ブックマークやリジューム点などである。Clip 又は PlayList にマークをセットすることは、マークの時刻を示すタイムスタンプをマークリストに追加することにより行われ

る。また、マークを削除することは、マークリストの中から、そのマークのタイムスタンプを除去することである。したがって、マークの設定や削除により、AVストリームは何の変更もされない。

ClipMarkの別のフォーマットとして、ClipMarkが参照するピクチャをAVストリームの中でのアドレスベースで指定するようにしてもよい。Clipにマークをセットすることは、マーク点のピクチャを示すアドレスベースの情報をマークリストに追加することにより行われる。また、マークを削除することは、マークリストの中から、そのマーク点のピクチャを示すアドレスベースの情報を除去することである。したがって、マークの設定や削除により、AVストリームは何の変更もされない。

次にサムネイルについて説明する。サムネイルは、Volume、PlayList及びClipに付加される静止画である。サムネイルには、2つの種類があり、1つは、内容を表す代表画としてのサムネイルである。これは主としてユーザがカーソル（不図示）などを操作して見たいものを選択するためのメニュー画面で使われるものである。もう1つは、マークが指しているシーンを表す画像である。

Volumeと各PlayListは代表画を持つことができるようにする必要がある。Volumeの代表画は、ディスク（記録媒体100、以下、記録媒体100はディスク状のものであるとし、適宜、ディスクと記述する）を記録再生装置1の所定の場所にセットしたときに、そのディスクの内容を表す静止画を最初に表示する場合などに用いられることを想定している。PlayListの代表画は、PlayListを選択するメニュー画面において、PlayListの内容を表すための静止画として用いられることを想定している。

PlayListの代表画として、PlayListの最初の画像をサムネイル（代表画）にすることが考えられるが、必ずしも再生時刻0の先頭の画像が内容を表す上で最適な画像とは限らない。そこで、PlayListのサムネイルとして、任意の画像をユーザが設定できるようにする。以上Volumeを表す代表画としてのサムネイルと、PlayListを表す代表画としてのサムネイルの2種類のサムネイルをメニューサムネイルと称する。メニューサムネイルは頻繁に表示されるため、ディスクから高速に読み出される必要がある。このため、全てのメニューサムネイルを1つのファ

イルに格納することが効率的である。メニューサムネイルは、必ずしもボリューム内の動画から抜き出したピクチャである必要はなく、図10に示すように、パーソナルコンピュータやデジタルスチルカメラから取り込まれた画像でもよい。

一方、ClipとPlayListには、複数のマークを打てる必要があり、マーク位置の内容を知るためにマーク点の画像を容易に見ることができるようにする必要がある。このようなマーク点を表すピクチャをマークサムネイル (Mark Thumbnail s) と称する。したがって、マークサムネイルの元となる画像は、外部から取り込んだ画像よりも、マーク点の画像を抜き出したものが主となる。

図11は、PlayListに付けられるマークと、そのマークサムネイルの関係について示す図であり、図12は、Clipに付けられるマークと、そのマークサムネイルの関係について示す図である。マークサムネイルは、メニューサムネイルと異なり、PlayListの詳細を表すときに、サブメニュー等で使われるため、短いアクセス時間で読み出されるようなことは要求されない。そのため、サムネイルが必要になる度に、記録再生装置1がファイルを開き、そのファイルの一部を読み出すことで多少時間がかかっても、問題にはならない。

また、ボリューム内に存在するファイル数を減らすために、全てのマークサムネイルは1つのファイルに格納するのがよい。PlayListはメニューサムネイル1つと複数のマークサムネイルを有することができるが、Clipは直接ユーザが選択する必要性がない（通常、PlayList経由で指定する）ため、メニューサムネイルを設ける必要はない。

図13は、上述したことを考慮した場合のメニューサムネイル、マークサムネイル、PlayList及びClipの関係について示した図である。メニューサムネイルファイルには、PlayList毎に設けられたメニューサムネイルがファイルされている。メニューサムネイルファイルには、ディスクに記録されているデータの内容を代表するボリュームサムネイルが含まれている。マークサムネイルファイルは、各PlayList毎と各Clip毎に作成されたサムネイルがファイルされている。

次に、CPI (Characteristic Point Information) について説明する。CPIは、Clipインフォメーションファイルに含まれるデータであり、主に、それはClipへのアクセスポイントのタイムスタンプが与えられたとき、Clip AV stream fileの

中でデータの読出を開始すべきデータアドレスを見つけるために用いられる。本実施の形態では、2種類のCPIを用いる。1つは、EP_mapであり、もう1つは、TU_mapである。

EP_mapは、エントリポイント(EP)データのリストであり、それはエレメンタリストリーム及びトランスポートストリームから抽出されたものである。これは、AVストリームの中でデコードを開始すべきエントリポイントの場所を見つけるためのアドレス情報を持つ。1つのEPデータは、プレゼンテーションタイムスタンプ(PTS)と、そのPTSに対応するアクセスユニットのAVストリームの中のデータアドレスの対で構成される。

EP_mapは、主に2つの目的のために使用される。第1に、PlayListの中でプレゼンテーションタイムスタンプによって参照されるアクセスユニットのAVストリームの中のデータアドレスを見つけるために使用される。第2に、ファーストフォワード再生やファーストリバース再生のために使用される。記録再生装置1が、入力AVストリームを記録する場合、そのストリームのシンタクスを解析することができるとき、EP_mapが作成され、ディスクに記録される。

TU_mapは、デジタルインタフェースを通して入力されるトランスポートパケットの到着時刻に基づいたタイムユニット(TU)データのリストを持つ。これは、到着時刻ベースの時間とAVストリームの中のデータアドレスとの関係を与える。記録再生装置1が、入力AVストリームを記録する場合、そのストリームのシンタクスを解析することができないとき、TU_mapが作成され、ディスクに記録される。

STCInfoは、MPEG-2トランスポートストリームをストアしているAVストリームファイルの中にあるSTCの不連続点情報をストアする。

AVストリームがSTCの不連続点を持つ場合、そのAVストリームファイルの中で同じ値のPTSが現れるかもしれない。そのため、AVストリーム上のある時刻をPTSベースで指す場合、アクセスポイントのPTSだけではそのポイントを特定するためには不十分である。さらに、そのPTSを含むところの連続なSTC区間のインデックスが必要である。連続なSTC区間を、このフォーマットではSTC-sequenceと呼び、そのインデックスをSTC-sequence-idと呼ぶ。STC-sequenceの情報は、Clip Information fileのSTCInfoで定義される。

STC-sequence-idは、EP_mapを持つAVストリームファイルで使用するものであり、TU_mapを持つAVストリームファイルではオプションである。

プログラムは、エレメンタリストリームの集まりであり、これらのストリームの同期再生のために、ただ1つのシステムタイムベースを共有するものである。

再生装置（図1の記録再生装置1）にとって、AVストリームのデコードに先立ち、そのAVストリームの内容がわかることは有用である。例えば、ビデオやオーディオのエレメンタリストリームを伝送するトランスポートパケットのPIDの値や、ビデオやオーディオのコンポーネント種類（例えば、HDTVのビデオとMPEG-2 AACのオーディオストリームなど）などの情報である。この情報はAVストリームを参照するところのPlayListの内容をユーザに説明するメニュー画面を作成するのに有用であるし、また、AVストリームのデコードに先だって、再生装置のAVデコーダ及びデマルチプレクサの初期状態をセットするために役立つ。

この理由のために、Clip Information fileは、プログラムの内容を説明するためのProgramInfoを持つ。

MPEG-2トランスポートストリームをストアしているAVストリームファイルは、ファイルの中でプログラム内容が変化するかもしれない。例えば、ビデオエレメンタリストリームを伝送するところのトランスポートパケットのPIDが変化したり、ビデオストリームのコンポーネント種類がSDTVからHDTVに変化するなどである。

ProgramInfoは、AVストリームファイルの中でのプログラム内容の変化点の情報をストアする。AVストリームファイルの中で、このフォーマットで定めるところのプログラム内容が一定である区間をprogram-sequenceと呼ぶ。

program-sequenceは、EP_mapを持つAVストリームファイルで使用するものであり、TU_mapを持つAVストリームファイルではオプションである。

本実施の形態では、セルフエンコードのストリームフォーマット（SESF）を定義する。SESFは、アナログ入力信号を符号化する目的、及びデジタル入力信号（例えばDV）をデコードしてからMPEG-2トランスポートストリームに符号化する場合に用いられる。

SESFは、MPEG-2トランスポートストリーム及びAVストリームについてのエレメンタリストリームの符号化制限を定義する。記録再生装置1が、SESFストリーム

をエンコードし、記録する場合、EP_mapが作成され、ディスクに記録される。

デジタル放送のストリームは、次に示す方式のうちのいずれかが用いられて記録媒体100に記録される。まず、デジタル放送のストリームをSESFストリームにトランスコーディングする。この場合、記録されたストリームは、SESFに準拠しなければならない。この場合、EP_mapが作成されて、ディスクに記録されなければならない。

あるいは、デジタル放送ストリームを構成するエレメンタリストリームを新しいエレメンタリストリームにトランスコーディングし、そのデジタル放送ストリームの規格化組織が定めるストリームフォーマットに準拠した新しいトランスポートストリームに再多重化する。この場合、EP_mapが作成されて、ディスクに記録されなければならない。

例えば、入力ストリームがISDB（日本のデジタルBS放送の規格名称）準拠のMP EG-2トランスポートストリームであり、それがHDTVビデオストリームとMPEG AACオーディオストリームを含むとする。HDTVビデオストリームをSDTVビデオストリームにトランスコーディングし、そのSDTVビデオストリームとオリジナルのAACオーディオストリームをTSに再多重化する。SDTVストリームと記録されるトランスポートストリームは、共にISDBフォーマットに準拠しなければならない。

デジタル放送のストリームが、記録媒体100に記録される際の他の方式として、入力トランスポートストリームをトランスペアレントに記録する（入力トランスポートストリームを何も変更しないで記録する）場合であり、そのときにEP_mapが作成されてディスクに記録される。

あるいは入力トランスポートストリームをトランスペアレントに記録する（入力トランスポートストリームを何も変更しないで記録する）場合であり、そのときにTU_mapが作成されてディスクに記録される。

次にディレクトリとファイルについて説明する。以下、記録再生装置1をDVR（Digital Video Recording）と適宜記述する。図14はディスク上のディレクトリ構造の一例を示す図である。DVRのディスク上に必要なディレクトリは、図14に示したように、“DVR”ディレクトリを含むrootディレクトリ、“PLAYLIST”ディレクトリ、“CLIPINF”ディレクトリ、“M2TS”ディレクトリ、及び“DATA”ディレクトリ

を含む”DVR”ディレクトリである。rootディレクトリの下に、これら以外のディレクトリを作成されるようにしてもよいが、それらは、本実施の形態のアプリケーションフォーマットでは、無視されたとする。

”DVR”ディレクトリの下には、DVRアプリケーションフォーマットによって規定される全てのファイルとディレクトリがストアされる。”DVR”ディレクトリは、4個のディレクトリを含む。”PLAYLIST”ディレクトリの下には、Real PlayListとVirtual PlayListのデータベースファイルが置かれる。このディレクトリは、PlayListが1つもなくても存在する。

”CLIPINF”ディレクトリの下には、Clipのデータベースが置かれる。このディレクトリも、Clipが1つもなくても存在する。”M2TS”ディレクトリの下には、AVストリームファイルが置かれる。このディレクトリは、AVストリームファイルが1つもなくても存在する。”DATA”ディレクトリは、デジタルTV放送などのデータ放送のファイルがストアされる。

”DVR”ディレクトリは、次に示すファイルをストアする。”info.dvr”ファイルは、DVRディレクトリの下に作られ、アプリケーションレイヤの全体的な情報をストアする。DVRディレクトリの下には、ただ1つのinfo.dvrがなければならない。ファイル名は、info.dvrに固定されたとする。”menu.thmb”ファイルは、メニューサムネイル画像に関連する情報をストアする。DVRディレクトリの下には、0又は1つのメニューサムネイルがなければならない。ファイル名は、menu.thmbに固定されたとする。メニューサムネイル画像が1つもない場合、このファイルは、存在しなくてもよい。

”mark.thmb”ファイルは、マークサムネイル画像に関連する情報をストアする。DVRディレクトリの下には、0又は1つのマークサムネイルがなければならない。ファイル名は、mark.thmbに固定されたとする。メニューサムネイル画像が1つもない場合、このファイルは、存在しなくてもよい。

”PLAYLIST”ディレクトリは、2種類のPlayListファイルをストアするものであり、それらは、Real PlayListとVirtual PlayListである。”xxxxx.rpls”ファイルは、1つのReal PlayListに関連する情報をストアする。それぞれのReal PlayList毎に、1つのファイルが作られる。ファイル名は、”xxxxx.rpls”である。ここ

で、“xxxxx”は、5個の0乃至9まで数字である。ファイル拡張子は、“rpls”でなければならないとする。

“yyyyy.vpls”ファイルは、1つのVirtual PlayListに関連する情報をストアする。それぞれのVirtual PlayList毎に、1つのファイルが作られる。ファイル名は、“yyyyy.vpls”である。ここで、“yyyyy”は、5個の0乃至9まで数字である。ファイル拡張子は、“vpls”でなければならないとする。

“CLIPINF”ディレクトリは、それぞれのAVストリームファイルに対応して、1つのファイルをストアする。“zzzzz.clpi”ファイルは、1つのAVストリームファイル (Clip AV stream file又はBridge-Clip AV stream file) に対応するClip Information fileである。ファイル名は、“zzzzz.clpi”であり、“zzzzz”は、5個の0乃至9までの数字である。ファイル拡張子は、“clpi”でなければならないとする。

“M2TS”ディレクトリは、AVストリームのファイルをストアする。“zzzzz.m2ts”ファイルは、DVRシステムにより扱われるAVストリームファイルである。これは、Clip AV stream file又はBridge-Clip AV streamである。ファイル名は、“zzzzz.m2ts”であり、“zzzzz”は、5個の0乃至9までの数字である。ファイル拡張子は、“m2ts”でなければならないとする。

“DATA”ディレクトリは、データ放送から伝送されるデータをストアするものであり、データとは、例えば、XML fileやMHEGファイルなどである。

次に、各ディレクトリ（ファイル）のシンタクスとセマンティクスを説明する。まず、“info.dvr”ファイルについて説明する。図15は、“info.dvr”ファイルのシンタクスを示す図である。“info.dvr”ファイルは、3個のオブジェクトから構成され、それらは、DVRVolume()、TableOfPlayLists()及びMakersPrivateData()である。

図15に示したinfo.dvrのシンタクスについて説明するに、TableOfPlayLists_Start_addressは、info.dvrファイルの先頭のバイトからの相対バイト数を単位として、TableOfPlayList()の先頭アドレスを示す。相対バイト数は0からカウントされる。

MakersPrivateData_Start_addressは、info.dvrファイルの先頭のバイトからの

相対バイト数を単位として、MakersPrivateData()の先頭アドレスを示す。相対バイト数は0からカウントされる。padding_word (パディングワード) は、info.dvrのシンタクスに従って挿入される。N1とN2は、0又は任意の正の整数である。それぞれのパディングワードは、任意の値を取るようにしてもよい。

DVRVolume()は、ボリューム (ディスク) の内容を記述する情報をストアする。図16は、DVRVolume()のシンタクスを示す図である。図16に示したDVR Volume()のシンタクスを説明するに、version_numberは、このDVRVolume()のバージョンナンバを示す4個のキャラクタ文字を示す。version_numberは、ISO 646に従って、"0045"と符号化される。

lengthは、このlengthフィールドの直後からDVRVolume()の最後までのDVRVolume()のバイト数を示す32ビットの符号なし整数で表される。

ResumeVolume()は、ボリュームの中で最後に再生したReal PlayList又はVirtual PlayListのファイル名を記憶している。ただし、Real PlayList又はVirtual PlayListの再生をユーザが中断したときの再生位置は、PlayListMark()において定義されるresume-markにストアされる (図42、図43)。

図17は、ResumeVolume()のシンタクスを示す図である。図17に示したResumeVolume()のシンタクスを説明するに、valid_flagは、この1ビットのフラグが1にセットされている場合、resume_PlayList_nameフィールドが有効であることを示し、このフラグが0にセットされている場合、resume_PlayList_nameフィールドが無効であることを示す。

resume_PlayList_nameの10バイトのフィールドは、リジュームされるべきReal PlayList又はVirtual PlayListのファイル名を示す。

図16に示したDVRVolume()のシンタクスの中の、UIAppInfoVolumeは、ボリュームについてのユーザインタフェースアプリケーションのパラメータをストアする。図18は、UIAppInfoVolumeのシンタクスを示す図であり、そのセマンティクスを説明するに、character_setの8ビットのフィールドは、Volume_nameフィールドに符号化されているキャラクタ文字の符号化方法を示す。その符号化方法は、図19に示される値に対応する。

name_lengthの8ビットフィールドは、Volume_nameフィールドの中に示される

ボリューム名のバイト長を示す。Volume_nameのフィールドは、ボリュームの名称を示す。このフィールドの中の左からname_length数のバイト数が、有効なキャラクタ文字であり、それはボリュームの名称を示す。Volume_nameフィールドの中で、それら有効なキャラクタ文字の後の値は、どんな値が入っていてもよい。

Volume_protect_flagは、ボリュームの中のコンテンツを、ユーザに制限することなしに見せてよいかどうかを示すフラグである。このフラグが1にセットされている場合、ユーザが正しくPIN番号（パスワード）を入力できたときだけ、そのボリュームのコンテンツを、ユーザに見せること（再生されること）が許可される。このフラグが0にセットされている場合、ユーザがPIN番号を入力しなくても、そのボリュームのコンテンツを、ユーザに見せることが許可される。

最初に、ユーザが、ディスクをプレーヤへ挿入した時点において、もしこのフラグが0にセットされているか、あるいはこのフラグが1にセットされていてもユーザがPIN番号を正しく入力できたならば、記録再生装置1は、そのディスクの中のPlaylistの一覧を表示させる。それぞれのPlaylistの再生制限は、Volume_protect_flagとは無関係であり、それはUIAppInfoPlaylist()の中に定義されるplayback_control_flagによって示される。

PINは、4個の0乃至9までの数字で構成され、それぞれの数字は、ISO/IEC 646に従って符号化される。ref_thumbnail_indexのフィールドは、ボリュームに付加されるサムネイル画像の情報を示す。ref_thumbnail_indexフィールドが、0xFFFFでない値の場合、そのボリュームにはサムネイル画像が付加されており、そのサムネイル画像は、menu.thumファイルの中にストアされている。その画像は、menu.thumファイルの中でref_thumbnail_indexの値を用いて参照される。ref_thumbnail_indexフィールドが、0xFFFFである場合、そのボリュームにはサムネイル画像が付加されていないことを示す。

次に図15に示したinfo.dvrのシンタクス内のTableOfPlayLists()について説明する。TableOfPlayLists()は、Playlist (Real PlaylistとVirtual Playlist)のファイル名をストアする。ボリュームに記録されている全てのPlaylistファイルは、TableOfPlayLists()の中に含まれる。TableOfPlayLists()は、ボリュームの中のPlaylistのデフォルトの再生順序を示す。

図 20 は、TableOfPlayLists()のシンタクスを示す図であり、そのシンタクスについて説明するに、TableOfPlayListsのversion_numberは、このTableOfPlayListsのバージョンナンバを示す4個のキャラクタ文字を示す。version_numberは、ISO 646に従って、“0045”と符号化されなければならない。

lengthは、このlengthフィールドの直後からTableOfPlayLists()の最後までのTableOfPlayLists()のバイト数を示す32ビットの符号なしの整数である。number_of_PlayListsの16ビットのフィールドは、Playlist_file_nameを含むfor-loopのループ回数を示す。この数字は、ボリュームに記録されているPlaylistの数に等しくなければならない。Playlist_file_nameの10バイトの数字は、Playlistのファイル名を示す。

図 21 は、TableOfPlayLists()のシンタクスの別の構成を示す図である。図 21 に示したシンタクスは、図 20 に示したシンタクスに、UIAppInfoPlaylist（後述）を含ませた構成とされている。このように、UIAppInfoPlaylistを含ませた構成とすることで、TableOfPlayListsを読み出すだけで、メニュー画面を作成することが可能となる。ここでは、図 20 に示したシンタクスを用いるとして以下の説明をする。

図 15 に示したinfo.dvrのシンタクス内のMakersPrivateDataについて説明する。MakersPrivateDataは、記録再生装置1のメーカーが、各社の特別なアプリケーションのために、MakersPrivateData()の中にメーカーのプライベートデータを挿入できるように設けられている。各メーカーのプライベートデータは、それを定義したメーカーを識別するために標準化されたmaker_IDを持つ。MakersPrivateData()は、1つ以上のmaker_IDを含んでもよい。

所定のメーカーが、プライベートデータを挿入したいときに、既に他のメーカーのプライベートデータがMakersPrivateData()に含まれていた場合、他のメーカーは、既にある古いプライベートデータを消去するのではなく、新しいプライベートデータをMakersPrivateData()の中に追加するようにする。このように、本実施の形態においては、複数のメーカーのプライベートデータが、1つのMakersPrivateData()に含まれることが可能であるようにする。

図 22 は、MakersPrivateDataのシンタクスを示す図である。図 22 に示したM

akersPrivateDataのシンタクスについて説明するに、version_numberは、このMakersPrivateData()のバージョンナンバを示す4個のキャラクタ文字を示す。version_numberは、ISO 646に従って、“0045”と符号化されなければならない。lengthは、このlengthフィールドの直後からMakersPrivateData()の最後までのMakersPrivateData()のバイト数を示す32ビットの符号なし整数を示す。

mpd_blocks_start_addressは、MakersPrivateData()の先頭のバイトからの相対バイト数を単位として、最初のmpd_block()の先頭バイトアドレスを示す。相対バイト数は0からカウントされる。number_of_maker_entriesは、MakersPrivateData()の中に含まれているメーカプライベートデータのエントリ数を与える16ビットの符号なし整数である。MakersPrivateData()の中に、同じmaker_IDの値を持つメーカプライベートデータが2個以上存在してはならない。

mpd_block_sizeは、1024バイトを単位として、1つのmpd_blockの大きさを与える16ビットの符号なし整数である。例えば、mpd_block_size=1ならば、それは1つのmpd_blockの大きさが1024バイトであることを示す。number_of_mpd_blocksは、MakersPrivateData()の中に含まれるmpd_blockの数を与える16ビットの符号なし整数である。maker_IDは、そのメーカプライベートデータを作成したDVRシステムの製造メーカを示す16ビットの符号なし整数である。maker_IDに符号化される値は、このDVRフォーマットのライセンサによって指定される。

maker_model_codeは、そのメーカプライベートデータを作成したDVRシステムのモデルナンバコードを示す16ビットの符号なし整数である。maker_model_codeに符号化される値は、このフォーマットのライセンスを受けた製造メーカによって設定される。start_mpd_block_numberは、そのメーカプライベートデータが開始されるmpd_blockの番号を示す16ビットの符号なし整数である。メーカプライベートデータの先頭データは、mpd_blockの先頭にアラインされなければならない。start_mpd_block_numberは、mpd_blockのfor-loopの中の変数jに対応する。

mpd_lengthは、バイト単位でメーカプライベートデータの大きさを示す32ビットの符号なし整数である。mpd_blockは、メーカプライベートデータがストアされる領域である。MakersPrivateData()の中の全てのmpd_blockは、同じサイズでなければならない。

次に、Real PlayList fileとVirtual PlayList fileについて、換言すれば、xxxx.rplsとyyyyy.vplsについて説明する。図23は、xxxxx.rpls (Real PlayList) 又はyyyyy.vpls (Virtual PlayList) のシンタクスを示す図である。xxxxx.rplsとyyyyy.vplsは、同一のシンタクス構成をもつ。xxxxx.rplsとyyyyy.vplsは、それぞれ、3個のオブジェクトから構成され、それらは、PlayList()、PlayListMark()及びMakersPrivateData()である。

PlayListMark_Start_addressは、PlayListファイルの先頭のバイトからの相対バイト数を単位として、PlayListMark()の先頭アドレスを示す。相対バイト数は0からカウントされる。

MakersPrivateData_Start_addressは、PlayListファイルの先頭のバイトからの相対バイト数を単位として、MakersPrivateData()の先頭アドレスを示す。相対バイト数は0からカウントされる。

padding_word (パディングワード) は、PlayListファイルのシンタクスに従って挿入され、N1とN2は、0又は任意の正の整数である。それぞれのパディングワードは、任意の値を取るようにしてもよい。

ここで、既に、簡便に説明したが、PlayListについてさらに説明する。ディスク内にある全てのReal PlayListによって、Bridge-Clip (後述) を除く全てのClipの中の再生区間が参照されていなければならない。かつ、2つ以上のReal PlayListが、それらのPlayItemで示される再生区間を同一のClipの中でオーバーラップさせてはならない。

図24A, 図24B, 図24Cを参照してさらに説明するに、図24Aに示したように、全てのClipは、対応するReal PlayListが存在する。この規則は、図24Bに示したように、編集作業が行われた後においても守られる。したがって、全てのClipは、どれかしらのReal PlayListを参照することにより、必ず視聴することが可能である。

図24Cに示したように、Virtual PlayListの再生区間は、Real PlayListの再生区間又はBridge-Clipの再生区間の中に含まれていなければならない。どのVirtual PlayListにも参照されないBridge-Clipがディスクの中に存在してはならない。

Real Playlistは、PlayItemのリストを含むが、SubPlayItemを含んではならない。Virtual Playlistは、PlayItemのリストを含み、Playlist()の中に示されるCPI_typeがEP_map typeであり、かつPlaylist_typeが0（ビデオとオーディオを含むPlaylist）である場合、Virtual Playlistは、1つのSubPlayItemを含むことができる。本実施の形態におけるPlaylist()では、SubPlayItemはオーディオのアフレコの目的にだけに使用される、そして、1つのVirtual Playlistが持つSubPlayItemの数は、0又は1でなければならない。

次に、Playlistについて説明する。図25は、Playlistのシンタクスを示す図である。図25に示したPlaylistのシンタクスを説明するに、version_numberは、このPlaylist()のバージョンナンバを示す4個のキャラクタ文字である。version_numberは、ISO 646に従って、“0045”と符号化されなければならない。lengthは、このlengthフィールドの直後からPlaylist()の最後まで Playlist()のバイト数を示す32ビットの符号なし整数である。Playlist_typeは、このPlaylistのタイプを示す8ビットのフィールドであり、その一例を図26に示す。

CPI_typeは、1ビットのフラグであり、PlayItem()及びSubPlayItem()によって参照されるClipのCPI_typeの値を示す。1つのPlaylistによって参照される全てのClipは、それらのCPI()の中に定義されるCPI_typeの値が同じでなければならない。number_of_PlayItemsは、Playlistの中にあるPlayItemの数を示す16ビットのフィールドである。

所定のPlayItem()に対応するPlayItem_idは、PlayItem()を含むfor-loopの中で、そのPlayItem()の現れる順番により定義される。PlayItem_idは、0から開始される。number_of_SubPlayItemsは、Playlistの中にあるSubPlayItemの数を示す16ビットのフィールドである。この値は、0又は1である。付加的なオーディオストリームのパス（オーディオストリームパス）は、サブパスの一種である。

次に、図25に示したPlaylistのシンタクスのUIAppInfoPlaylistについて説明する。UIAppInfoPlaylistは、Playlistについてのユーザインタフェースアプリケーションのパラメータをストアする。図27は、UIAppInfoPlaylistのシンタクスを示す図である。図27に示したUIAppInfoPlaylistのシンタクスを説明するに、character_setは、8ビットのフィールドであり、Playlist_nameフィールドに符

号化されているキャラクタ文字の符号化方法を示す。その符号化方法は、図 19 に示したテーブルに準拠する値に対応する。

name_lengthは、8ビットフィールドであり、PlayList_nameフィールドの中に示されるPlayList名のバイト長を示す。PlayList_nameのフィールドは、PlayListの名称を示す。このフィールドの中の左からname_length数のバイト数が、有効なキャラクタ文字であり、それはPlayListの名称を示す。PlayList_nameフィールドの中で、それら有効なキャラクタ文字の後の値は、どんな値が入っていてもよい。

record_time_and_dateは、PlayListが記録されたときの日時をストアする56ビットのフィールドである。このフィールドは、年／月／日／時／分／秒について、14個の数字を4ビットのBinary Coded Decimal (BCD) で符号化したものである。例えば、2001/12/23:01:02:03は、“0x20011223010203”と符号化される。

durationは、PlayListの総再生時間を時間／分／秒の単位で示した24ビットのフィールドである。このフィールドは、6個の数字を4ビットのBinary Coded Decimal (BCD) で符号化したものである。例えば、01:45:30は、“0x014530”と符号化される。

valid_periodは、PlayListが有効である期間を示す32ビットのフィールドである。このフィールドは、8個の数字を4ビットのBinary Coded Decimal (BCD) で符号化したものである。例えば、記録再生装置1は、この有効期間の過ぎたPlayListを自動消去する、といったように用いられる。例えば、2001/05/07は、“0x20010507”と符号化される。

maker_IDは、そのPlayListを最後に更新したDVRプレーヤ（記録再生装置1）の製造者を示す16ビットの符号なし整数である。maker_IDに符号化される値は、DVRフォーマットのライセンスによって割り当てられる。maker_codeは、そのPlayListを最後に更新したDVRプレーヤのモデル番号を示す16ビットの符号なし整数である。maker_codeに符号化される値は、DVRフォーマットのライセンスを受けた製造者によって決められる。

playback_control_flagのフラグが1にセットされている場合、ユーザが正しくPIN番号を入力できた場合にだけ、そのPlayListは再生される。このフラグが0に

セットされている場合、ユーザがPIN番号を入力しなくても、ユーザは、そのPlaylistを視聴することができる。

write_protect_flagは、図28Aにテーブルを示すように、1にセットされている場合、write_protect_flagを除いて、そのPlaylistの内容は、消去及び変更されない。このフラグが0にセットされている場合、ユーザは、そのPlaylistを自由に消去及び変更できる。このフラグが1にセットされている場合、ユーザが、そのPlaylistを消去、編集又は上書きする前に、記録再生装置1はユーザに再確認するようなメッセージを表示させる。

write_protect_flagが0にセットされているReal Playlistが存在し、かつ、そのReal PlaylistのClipを参照するVirtual Playlistが存在し、そのVirtual Playlistのwrite_protect_flagが1にセットされていてもよい。ユーザが、Real Playlistを消去しようとする場合、記録再生装置1は、そのReal Playlistを消去する前に、上述のVirtual Playlistの存在をユーザに警告するか、あるいはそのReal Playlistを"Minimize"する。

is_played_flagは、図28Bに示すように、フラグが1にセットされている場合、そのPlaylistは、記録されてから一度は再生されたことを示し、0にセットされている場合、そのPlaylistは、記録されてから一度も再生されたことがないことを示す。

archiveは、図28Cに示すように、そのPlaylistがオリジナルであるか、コピーされたものであるかを示す2ビットのフィールドである。ref_thumbnail_indexのフィールドは、Playlistを代表するサムネイル画像の情報を示す。ref_thumbnail_indexフィールドが、0xFFFFでない値の場合、そのPlaylistには、Playlistを代表するサムネイル画像が付加されており、そのサムネイル画像は、menu.thumファイルの中にストアされている。その画像は、menu.thumファイルの中でref_thumbnail_indexの値を用いて参照される。ref_thumbnail_indexフィールドが、0xFFFFである場合、そのPlaylistには、Playlistを代表するサムネイル画像が付加されていない。

次にPlayItemについて説明する。1つのPlayItem()は、基本的に次のデータを含む。Clipのファイル名を指定するためのClip_Information_file_name、Clipの

再生区間を特定するためのIN_timeとOUT_timeのペア、PlayList()において定義されるCPI_typeがEP_map typeである場合、IN_timeとOUT_timeが参照するところのSTC_sequence_id、及び先行するPlayItemと現在のPlayItemとの接続の状態を示すところのConnection_Conditionである。

PlayListが2つ以上のPlayItemから構成されるとき、それらのPlayItemはPlayListのグローバル時間軸上に、時間のギャップ又はオーバーラップなしに一系列に並べられる。PlayList()において定義されるCPI_typeがEP_map typeであり、かつ現在のPlayItemがBridgeSequence()を持たないとき、そのPlayItemにおいて定義されるIN_timeとOUT_timeのペアは、STC_sequence_idによって指定される同じSTC連続区間上の時間を指していなければならない。そのような例を図29に示す。

図30は、PlayList()において定義されるCPI_typeがEP_map typeであり、かつ現在のPlayItemがBridgeSequence()を持つとき、次に説明する規則が適用される場合を示している。現在のPlayItemに先行するPlayItemのIN_time（図の中でIN_time1と示されているもの）は、先行するPlayItemのSTC_sequence_idによって指定されるSTC連続区間上の時間を指している。先行するPlayItemのOUT_time（図の中でOUT_time1と示されているもの）は、現在のPlayItemのBridgeSequenceInfo()の中で指定されるBridge-Clipの中の時間を指している。このOUT_timeは、後述する符号化制限に従っていなければならない。

現在のPlayItemのIN_time（図の中でIN_time2と示されているもの）は、現在のPlayItemのBridgeSequenceInfo()の中で指定されるBridge-Clipの中の時間を指している。このIN_timeも、後述する符号化制限に従っていなければならない。現在のPlayItemのOUT_time（図の中でOUT_time2と示されているもの）は、現在のPlayItemのSTC_sequence_idによって指定されるSTC連続区間上の時間を指している。

図31に示すように、PlayList()のCPI_typeがTU_map typeである場合、PlayItemのIN_timeとOUT_timeのペアは、同じClip AVストリーム上の時間を指している。

PlayItemのシンタクスは、図32に示すようになる。図32に示したPlayItemのシンタクスを説明するに、Clip_Information_file_nameのフィールドは、Clip Information fileのファイル名を示す。このClip Information fileのClipInfo

()において定義されるClip_stream_typeは、Clip AV streamを示していなければならない。

STC_sequence_idは、8ビットのフィールドであり、PlayItemが参照するSTC連続区間のSTC_sequence_idを示す。PlayList()の中で指定されるCPI_typeがTU_map typeである場合、この8ビットフィールドは何も意味を持たず、0にセットされる。IN_timeは、32ビットフィールドであり、PlayItemの再生開始時刻をストアする。IN_timeのセマンティクスは、図33に示すように、PlayList()において定義されるCPI_typeによって異なる。

OUT_timeは、32ビットフィールドであり、PlayItemの再生終了時刻をストアする。OUT_timeのセマンティクスは、図34に示すように、PlayList()において定義されるCPI_typeによって異なる。

Connection_Conditionは、図35に示したような先行するPlayItemと、現在のPlayItemとの間の接続状態を示す2ビットのフィールドである。図36A, 図36B, 図36C, 図36Dは、図35に示したConnection_Conditionの各状態について説明する図である。

次に、BridgeSequenceInfoについて、図37を参照して説明する。BridgeSequenceInfo()は、現在のPlayItemの付属情報であり、次に示す情報を持つ。Bridge-Clip AV streamファイルとそれに対応するClip Information file (図45) を指定するBridge_Clip_Information_file_nameを含む。

また、先行するPlayItemが参照するClip AV stream上のソースパケットのアドレスであり、このソースパケットに続いてBridge-Clip AV streamファイルの最初のソースパケットが接続される。このアドレスは、RSPN_exit_from_previous_Clipと称される。さらに現在のPlayItemが参照するClip AV stream上のソースパケットのアドレスであり、このソースパケットの前にBridge-Clip AV streamファイルの最後のソースパケットが接続される。このアドレスは、RSPN_enter_to_current_Clipと称される。

図37において、RSPN_arrival_time_discontinuityは、the Bridge-Clip AV streamファイルの中でアライバルタイムベースの不連続点があるところのソースパケットのアドレスを示す。このアドレスは、ClipInfo() (図46) の中におい

て定義される。

図 3 8 は、BridgeSequenceInfoのシンタクスを示す図である。図 3 8 に示した BridgeSequenceInfoのシンタクスを説明するに、Bridge_Clip_Information_file_nameのフィールドは、Bridge-Clip AV streamファイルに対応するClip Information fileのファイル名を示す。このClip Information fileのClipInfo()において定義されるClip_stream_typeは、'Bridge-Clip AV stream'を示していなければならない。

RSPN_exit_from_previous_Clipの 3 2 ビットフィールドは、先行するPlayItemが参照するClip AV stream上のソースパケットの相対アドレスであり、このソースパケットに続いてBridge-Clip AV streamファイルの最初のソースパケットが接続される。RSPN_exit_from_previous_Clipは、ソースパケット番号を単位とする大きさであり、先行するPlayItemが参照するClip AV streamファイルの最初のソースパケットからClipInfo()において定義されるoffset_SPNの値を初期値としてカウントされる。

RSPN_enter_to_current_Clipの 3 2 ビットフィールドは、現在のPlayItemが参照するClip AV stream上のソースパケットの相対アドレスであり、このソースパケットの前にBridge-Clip AV streamファイルの最後のソースパケットが接続される。RSPN_exit_from_previous_Clipは、ソースパケット番号を単位とする大きさであり、現在のPlayItemが参照するClip AV streamファイルの最初のソースパケットからClipInfo()において定義されるoffset_SPNの値を初期値としてカウントされる。

次に、SubPlayItemについて、図 3 9 を参照して説明する。SubPlayItem()の使用は、PlayList()のCPI_typeがEP_map typeである場合にだけに許される。本実施の形態においては、SubPlayItemはオーディオのアフレコの目的のためだけに使用されるとする。SubPlayItem()は、次に示すデータを含む。まず、PlayListの中のsub pathが参照するClipを指定するためのClip_Information_file_nameを含む。

また、Clipの中のsub pathの再生区間を指定するためのSubPath_IN_timeとSubPath_OUT_timeを含む。さらに、main pathの時間軸上でsub pathが再生開始する時刻を指定するためのsync_PlayItem_idとsync_start_PTS_of_PlayItemを含む。

sub pathに参照されるオーディオのClip AV streamは、STC不連続点（システムタイムベースの不連続点）を含んではならない。sub pathに使われるClipのオーディオサンプルのクロックは、main pathのオーディオサンプルのクロックにロックされている。

図40は、SubPlayItemのシンタクスを示す図である。図40に示したSubPlayItemのシンタクスを説明するに、Clip_Information_file_nameのフィールドは、Clip Information fileのファイル名を示し、それはPlayListの中でsub pathによって使用される。このClip Information fileのClipInfo()において定義されるClip_stream_typeは、Clip AV streamを示していなければならない。

SubPath_typeの8ビットのフィールドは、sub pathのタイプを示す。ここでは、図41に示すように、'0x00'しか設定されておらず、他の値は、将来のために確保されている。

sync_PlayItem_idの8ビットのフィールドは、main pathの時間軸上でsub pathが再生開始する時刻が含まれるPlayItemのPlayItem_idを示す。所定のPlayItemに対応するPlayItem_idの値は、PlayList()において定義される（図25参照）。

sync_start_PTS_of_PlayItemの32ビットのフィールドは、main pathの時間軸上でsub pathが再生開始する時刻を示し、sync_PlayItem_idで参照されるPlayItem上のPTS (Presentation Time Stamp) の上位32ビットを示す。SubPath_IN_timeの32ビットフィールドは、sub pathの再生開始時刻をストアする。SubPath_IN_timeは、sub pathの中で最初のプレゼンテーションユニットに対応する33ビット長のPTSの上位32ビットを示す。

SubPath_OUT_timeの32ビットフィールドは、sub pathの再生終了時刻をストアする。SubPath_OUT_timeは、次式によって算出されるPresentation_end_TSの値の上位32ビットを示す。

$$\text{Presentation_end_TS} = \text{PTS_out} + \text{AU_duration}$$

ここで、PTS_outは、SubPathの最後のプレゼンテーションユニットに対応する33ビット長のPTSである。AU_durationは、SubPathの最後のプレゼンテーションユニットの90kHz単位の表示期間である。

次に、図23に示したxxxxx.rplsとyyyyy.vplsのシンタクス内のPlayListMark

()について説明する。PlayListについてのマーク情報は、このPlayListMarkにストアされる。図42は、PlayListMarkのシンタクスを示す図である。図42に示したPlayListMarkのシンタクスについて説明するに、version_numberは、このPlayListMark()のバージョンナンバを示す4個のキャラクタ文字である。version_numberは、ISO 646に従って、"0045"と符号化されなければならない。

lengthは、このlengthフィールドの直後からPlayListMark()の最後まででのPlayListMark()のバイト数を示す32ビットの符号なし整数である。number_of_PlayList_marksは、PlayListMarkの中にストアされているマークの個数を示す16ビットの符号なし整数である。number_of_PlayList_marksは、0であってもよい。mark_typeは、マークのタイプを示す8ビットのフィールドであり、図43に示すテーブルに従って符号化される。

mark_time_stampの32ビットフィールドは、マークが指定されたポイントを示すタイムスタンプをストアする。mark_time_stampのセマンティクスは、図44に示すように、PlayList()において定義されるCPI_typeによって異なる。PlayItem_idは、マークが置かれているところのPlayItemを指定する8ビットのフィールドである。所定のPlayItemに対応するPlayItem_idの値は、PlayList()において定義される(図25参照)。

character_setの8ビットのフィールドは、mark_nameフィールドに符号化されているキャラクタ文字の符号化方法を示す。その符号化方法は、図19に示した値に対応する。name_lengthの8ビットフィールドは、mark_nameフィールドの中に示されるマーク名のバイト長を示す。mark_nameのフィールドは、マークの名称を示す。このフィールドの中の左からname_length数のバイト数が、有効なキャラクタ文字であり、それはマークの名称を示す。mark_nameフィールドの中で、それら有効なキャラクタ文字の後の値は、どのような値が設定されてもよい。

ref_thumbnail_indexのフィールドは、マークに付加されるサムネイル画像の情報を示す。ref_thumbnail_indexフィールドが、0xFFFFでない値の場合、そのマークにはサムネイル画像が付加されており、そのサムネイル画像は、mark.thmbファイルの中にストアされている。その画像は、mark.thmbファイルの中でref_thumbnail_indexの値を用いて参照される(後述)。ref_thumbnail_indexフィールドが、

0xFFFFである場合、そのマークにはサムネイル画像が付加されていないことを示す。

次に、Clip Information fileについて説明する。zzzzz.clpi (Clip Information fileファイル) は、図 4 5 に示すように 6 個のオブジェクトから構成される。それらは、ClipInfo()、STC_Info()、ProgramInfo()、CPI()、ClipMark()及びMakersPrivateData()である。AVストリーム (Clip AVストリーム又はBridge-Clip AV stream) とそれに対応するClip Informationファイルは、同じ数字列の”zzzzz”が使用される。

図 4 5 に示したzzzzz.clpi (Clip Information fileファイル) のシンタクスについて説明するに、ClipInfo_Start_addressは、zzzzz.clpiファイルの先頭のバイトからの相対バイト数を単位として、ClipInfo()の先頭アドレスを示す。相対バイト数は 0 からカウントされる。

STC_Info_Start_addressは、zzzzz.clpiファイルの先頭のバイトからの相対バイト数を単位として、STC_Info()の先頭アドレスを示す。相対バイト数は 0 からカウントされる。ProgramInfo_Start_addressは、zzzzz.clpiファイルの先頭のバイトからの相対バイト数を単位として、ProgramInfo()の先頭アドレスを示す。相対バイト数は 0 からカウントされる。CPI_Start_addressは、zzzzz.clpiファイルの先頭のバイトからの相対バイト数を単位として、CPI()の先頭アドレスを示す。相対バイト数は 0 からカウントされる。

ClipMark_Start_addressは、zzzzz.clpiファイルの先頭のバイトからの相対バイト数を単位として、ClipMark()の先頭アドレスを示す。相対バイト数は 0 からカウントされる。MakersPrivateData_Start_addressは、zzzzz.clpiファイルの先頭のバイトからの相対バイト数を単位として、MakersPrivateData()の先頭アドレスを示す。相対バイト数は 0 からカウントされる。padding_word (パディングワード) は、zzzzz.clpiファイルのシンタクスに従って挿入される。N 1、N 2、N 3、N 4 及びN 5 は、0 又は任意の正の整数でなければならない。それぞれのパディングワードは、任意の値がとられるようにしてもよい。

次に、ClipInfoについて説明する。図 4 6 は、ClipInfoのシンタクスを示す図である。ClipInfo()は、それに対応するAVストリームファイル (Clip AVストリー

ム又はBridge-Clip AVストリームファイル)の属性情報をストアする。

図4 6 に示したClipInfoのシンタクスについて説明するに、version_numberは、このClipInfo()のバージョンナンバを示す4個のキャラクタ文字である。version_numberは、ISO 646に従って、“0045”と符号化されなければならない。lengthは、このlengthフィールドの直後からClipInfo()の最後までのClipInfo()のバイト数を示す32ビットの符号なし整数である。Clip_stream_typeの8ビットのフィールドは、図4 7 に示すように、Clip Informationファイルに対応するAVストリームのタイプを示す。それぞれのタイプのAVストリームのストリームタイプについては後述する。

offset_SPNの32ビットのフィールドは、AVストリーム (Clip AVストリーム又はBridge-Clip AVストリーム) ファイルの最初のソースパケットについてのソースパケット番号のオフセット値を与える。AVストリームファイルが最初にディスクに記録されるとき、このoffset_SPNは0でなければならない。

図4 8 に示すように、AVストリームファイルのはじめの部分が編集によって消去されたとき、offset_SPNは、0以外の値をとってもよい。本実施の形態では、offset_SPNを参照する相対ソースパケット番号 (相対アドレス) が、しばしば、RSPN_xxx (xxxは変形する。例えば、RSPN_EP_start) の形式でシンタクスの中に記述されている。相対ソースパケット番号は、ソースパケット番号を単位とする大きさであり、AVストリームファイルの最初のソースパケットからoffset_SPNの値を初期値としてカウントされる。

AVストリームファイルの最初のソースパケットから相対ソースパケット番号で参照されるソースパケットまでのソースパケットの数 (SPN_xxx) は、次式で算出される。

$$\text{SPN_xxx} = \text{RSPN_xxx} - \text{offset_SPN}$$

図4 8 に、offset_SPNが、4である場合の例を示す。

TS_recording_rateは、24ビットの符号なし整数であり、この値は、DVRドライブ (書込部22) へ又はDVRドライブ (読出部28) からのAVストリームの必要な入出力のビットレートを与える。record_time_and_dateは、Clipに対応するAVストリームが記録されたときの日時をストアする56ビットのフィールドであり、

年／月／日／時／分／秒について、14個の数字を4ビットのBinary Coded Decimal (BCD) で符号化したものである。例えば、2001/12/23:01:02:03は、“0x20011223010203”と符号化される。

durationは、Clipの総再生時間をアライバルタイムクロックに基づいた時間／分／秒の単位で示した24ビットのフィールドである。このフィールドは、6個の数字を4ビットのBinary Coded Decimal (BCD) で符号化したものである。例えば、01:45:30は、“0x014530”と符号化される。

time_controlled_flagのフラグは、AVストリームファイルの記録モードを示す。このtime_controlled_flagが1である場合、記録モードは、記録してからの時間経過に対してファイルサイズが比例するようにして記録されるモードであることを示し、次式に示す条件を満たさなければならない。

$$\begin{aligned} TS_average_rate * 192 / 188 * (t - start_time) - \alpha &\leq size_clip(t) \\ &\leq TS_average_rate * 192 / 188 * (t - start_time) + \alpha \end{aligned}$$

ここで、TS_average_rateは、AVストリームファイルのトランスポートストリームの平均ビットレートをbytes/secondの単位で表したものである。

また、上式において、tは、秒単位で表される時間を示し、start_timeは、AVストリームファイルの最初のソースパケットが記録されたときの時刻であり、秒単位で表される。size_clip(t)は、時刻tにおけるAVストリームファイルのサイズをバイト単位で表したものであり、例えば、start_timeから時刻tまでに10個のソースパケットが記録された場合、size_clip(t)は10 * 192バイトである。 α は、TS_average_rateに依存する定数である。

time_controlled_flagが0にセットされている場合、記録モードは、記録の時間経過とAVストリームのファイルサイズが比例するように制御していないことを示す。例えば、これは入力トランスポートストリームをトランスベアレント記録する場合である。

TS_average_rateは、time_controlled_flagが1にセットされている場合、この24ビットのフィールドは、上式で用いているTS_average_rateの値を示す。time_controlled_flagが0にセットされている場合、このフィールドは、何も意味を持たず、0にセットされなければならない。例えば、可変ビットレートのトラン

スポーツストリームは、次に示す手順により符号化される。まずトランスポートレートをTS_recording_rateの値にセットする。次に、ビデオストリームを可変ビットレートで符号化する。そして、ヌルパケットを使用しないことによって、間欠的にトランスポートパケットを符号化する。

RSPN_arrival_time_discontinuityの32ビットフィールドは、Bridge-Clip AV streamファイル上でアライバルタイムベースの不連続が発生する場所の相対アドレスである。RSPN_arrival_time_discontinuityは、ソースパケット番号を単位とする大きさであり、Bridge-Clip AV streamファイルの最初のソースパケットからClipInfo()において定義されるoffset_SPNの値を初期値としてカウントされる。そのBridge-Clip AV streamファイルの中での絶対アドレスは、上述した

$$\text{SPN_xxx} = \text{RSPN_xxx} - \text{offset_SPN}$$

に基づいて算出される。

reserved_for_system_useの144ビットのフィールドは、システム用にリザーブされている。is_format_identifier_validのフラグが1であるとき、format_identifierのフィールドが有効であることを示す。is_original_network_ID_validのフラグが1である場合、original_network_IDのフィールドが有効であることを示す。is_transport_stream_ID_validのフラグが1である場合、transport_stream_IDのフィールドが有効であることを示す。is_service_ID_validのフラグが1である場合、service_IDのフィールドが有効であることを示す。

is_country_code_validのフラグが1であるとき、country_codeのフィールドが有効であることを示す。format_identifierの32ビットフィールドは、トランスポートストリームの中でregistration_descriptor (ISO/IEC13818-1で定義されている) が持つformat_identifierの値を示す。original_network_IDの16ビットフィールドは、トランスポートストリームの中で定義されているoriginal_network_IDの値を示す。transport_stream_IDの16ビットフィールドは、トランスポートストリームの中で定義されているtransport_stream_IDの値を示す。

service_IDの16ビットフィールドは、トランスポートストリームの中で定義されているservice_IDの値を示す。country_codeの24ビットのフィールドは、ISO3166によって定義されるカントリーコードを示す。それぞれのキャラクタ文字

は、ISO8859-1で符号化される。例えば、日本は"JPN"と表され、"0x4A 0x50 0x4E"と符号化される。stream_format_nameは、トランスポートストリームのストリーム定義をしているフォーマット機関の名称を示すISO-646の16個のキャラクタコードである。このフィールドの中の無効なバイトは、値'0xFF'がセットされる。

format_identifier、original_network_ID、transport_stream_ID、service_ID、country_code及びstream_format_nameは、トランスポートストリームのサービスプロバイダを示すものであり、これにより、オーディオやビデオストリームの符号化制限、SI（サービスインフォメーション）の規格やオーディオビデオストリーム以外のプライベートデータストリームのストリーム定義を認識することができる。これらの情報は、デコーダが、そのストリームをデコードできるか否か、そしてデコードできる場合にデコード開始前にデコーダシステムの初期設定を行うために用いることが可能である。

次に、STC_Infoについて説明する。ここでは、MPEG-2トランスポートストリームの中でSTCの不連続点（システムタイムベースの不連続点）を含まない時間区間をSTC_sequenceと称し、Clipの中で、STC_sequenceは、STC_sequence_idの値によって特定される。図50A、図50Bは、連続なSTC区間について説明する図である。同じSTC_sequenceの中で同じSTCの値は、決して現れない（ただし、後述するように、Clipの最大時間長は制限されている）。したがって、同じSTC_sequenceの中で同じPTSの値もまた、決して現れない。AVストリームが、N (N>0) 個のSTC不連続点を含む場合、Clipのシステムタイムベースは、(N+1)個のSTC_sequenceに分割される。

STC_Infoは、STCの不連続（システムタイムベースの不連続）が発生する場所のアドレスをストアする。図51を参照して説明するように、RSPN_STC_startが、そのアドレスを示し、最後のSTC_sequenceを除くk番目 (k>=0) のSTC_sequenceは、k番目のRSPN_STC_startで参照されるソースパケットが到着した時刻から始まり、(k+1)番目のRSPN_STC_startで参照されるソースパケットが到着した時刻で終わる。最後のSTC_sequenceは、最後のRSPN_STC_startで参照されるソースパケットが到着した時刻から始まり、最後のソースパケットが到着した時刻で終了する。

図52は、STC_Infoのシンタクスを示す図である。図52に示したSTC_Infoの

シンタクスについて説明するに、version_numberは、このSTC_Info()のバージョンナンバを示す4個のキャラクタ文字である。version_numberは、ISO 646に従って、“0045”と符号化されなければならない。

lengthは、このlengthフィールドの直後からSTC_Info()の最後までSTC_Info()のバイト数を示す32ビットの符号なし整数である。CPI()のCPI_typeがTU_map typeを示す場合、このlengthフィールドは0をセットしてもよい。CPI()のCPI_typeがEP_map typeを示す場合、num_of_STC_sequencesは1以上の値でなければならない。

num_of_STC_sequencesの8ビットの符号なし整数は、Clipの中でのSTC_sequenceの数を示す。この値は、このフィールドに続くfor-loopのループ回数を示す。所定のSTC_sequenceに対応するSTC_sequence_idは、RSPN_STC_startを含むfor-loopの中で、そのSTC_sequenceに対応するRSPN_STC_startの現れる順番により定義されるものである。STC_sequence_idは、0から開始される。

RSPN_STC_startの32ビットフィールドは、AVストリームファイル上でSTC_sequenceが開始するアドレスを示す。RSPN_STC_startは、AVストリームファイルの中でシステムタイムベースの不連続点が発生するアドレスを示す。RSPN_STC_startは、AVストリームの中で新しいシステムタイムベースの最初のPCRを持つソースパケットの相対アドレスとしてもよい。RSPN_STC_startは、ソースパケット番号を単位とする大きさであり、AVストリームファイルの最初のソースパケットからClipInfo()において定義されるoffset_SPNの値を初期値としてカウントされる。そのAV streamファイルの中での絶対アドレスは、既に上述した

$$\text{SPN_xxx} = \text{RSPN_xxx} - \text{offset_SPN}$$

により算出される。

次に、図45に示したzzzzz.clipのシンタクス内のProgramInfoについて説明する。図53を参照しながら説明するに、ここでは、Clipの中で次の特徴をもつ時間区間をprogram_sequenceと呼ぶ。まず、PCR_PIDの値が変わらない。次に、ビデオエレメンタリストリームの数が変化しない。また、それぞれのビデオストリームについてのPIDの値とそのVideoCodingInfoによって定義される符号化情報が変化しない。さらに、オーディオエレメンタリストリームの数が変化しない。また、

それぞれのオーディオストリームについてのPIDの値とそのAudioCodingInfoによって定義される符号化情報が変化しない。

program_sequenceは、同一の時刻において、ただ1つのシステムタイムベースを持つ。program_sequenceは、同一の時刻において、ただ1つのPMTを持つ。ProgramInfo()は、program_sequenceが開始する場所のアドレスをストアする。RSPN_program_sequence_startが、そのアドレスを示す。

図54は、ProgramInfoのシンタクスを示す図である。図54に示したProgramInfoのシンタクスを説明するに、version_numberは、このProgramInfo()のバージョンナンバを示す4個のキャラクタ文字である。version_numberは、ISO 646に従って、"0045"と符号化されなければならない。

lengthは、このlengthフィールドの直後からProgramInfo()の最後までのProgramInfo()のバイト数を示す32ビットの符号なし整数である。CPI()のCPI_typeがTU_map typeを示す場合、このlengthフィールドは0にセットされてもよい。CPI()のCPI_typeがEP_map typeを示す場合、number_of_programsは1以上の値でなければならない。

number_of_program_sequencesの8ビットの符号なし整数は、Clipの中でのprogram_sequenceの数を示す。この値は、このフィールドに続くfor-loopのループ回数を示す。Clipの中でprogram_sequenceが変化しない場合、number_of_program_sequencesは1をセットされなければならない。RSPN_program_sequence_startの32ビットフィールドは、AVストリームファイル上でプログラムシーケンスが開始する場所の相対アドレスである。

RSPN_program_sequence_startは、ソースパケット番号を単位とする大きさであり、AVストリームファイルの最初のソースパケットからClipInfo()において定義されるoffset_SPNの値を初期値としてカウントされる。そのAVストリームファイルの中での絶対アドレスは、

$$\text{SPN}_{xxx} = \text{RSPN}_{xxx} - \text{offset_SPN}$$

により算出される。シンタクスのfor-loopの中でRSPN_program_sequence_start値は、昇順に現れなければならない。

PCR_PIDの16ビットフィールドは、そのprogram_sequenceに有効なPCRフィー

ルドを含むトランスポートパケットのPIDを示す。number_of_videosの8ビットフィールドは、video_stream_PIDとVideoCodingInfo()を含むfor-loopのループ回数を示す。number_of_audiosの8ビットフィールドは、audio_stream_PIDとAudioCodingInfo()を含むfor-loopのループ回数を示す。video_stream_PIDの16ビットフィールドは、そのprogram_sequenceに有効なビデオストリームを含むトランスポートパケットのPIDを示す。このフィールドに続くVideoCodingInfo()は、そのvideo_stream_PIDで参照されるビデオストリームの内容を説明しなければならない。

audio_stream_PIDの16ビットフィールドは、そのprogram_sequenceに有効なオーディオストリームを含むトランスポートパケットのPIDを示す。このフィールドに続くAudioCodingInfo()は、そのaudio_stream_PIDで参照されるビデオストリームの内容を説明しなければならない。

なお、シンタクスのfor-loopの中でvideo_stream_PIDの値の現れる順番は、そのprogram_sequenceに有効なPMTの中でビデオストリームのPIDが符号化されている順番に等しくなければならない。また、シンタクスのfor-loopの中でaudio_stream_PIDの値の現れる順番は、そのprogram_sequenceに有効なPMTの中でオーディオストリームのPIDが符号化されている順番に等しくなければならない。

図55は、図54に示したProgramInfoのシンタクス内のVideoCodingInfoのシンタクスを示す図である。図55に示したVideoCodingInfoのシンタクスを説明するに、video_formatの8ビットフィールドは、図56に示すように、ProgramInfo()の中のvideo_stream_PIDに対応するビデオフォーマットを示す。

frame_rateの8ビットフィールドは、図57に示すように、ProgramInfo()の中のvideo_stream_PIDに対応するビデオのフレームレートを示す。display_aspect_ratioの8ビットフィールドは、図58に示すように、ProgramInfo()の中のvideo_stream_PIDに対応するビデオの表示アスペクト比を示す。

図59は、図54に示したProgramInfoのシンタクス内のAudioCodingInfoのシンタクスを示す図である。図59に示したAudioCodingInfoのシンタクスを説明するに、audio_codingの8ビットフィールドは、図60に示すように、ProgramInfo()の中のaudio_stream_PIDに対応するオーディオの符号化方法を示す。

audio_component_typeの8ビットフィールドは、図61に示すように、ProgramInfo()の中のaudio_stream_PIDに対応するオーディオのコンポーネントタイプを示す。sampling_frequencyの8ビットフィールドは、図62に示すように、ProgramInfo()の中のaudio_stream_PIDに対応するオーディオのサンプリング周波数を示す。

次に、図45に示したzzzzz.clipのシンタクス内のCPI (Characteristic Point Information) について説明する。CPIは、AVストリームの中の時間情報とそのファイルの中のアドレスとを関連付けるためにある。CPIには2つのタイプがあり、それらはEP_mapとTU_mapである。図63に示すように、CPI()の中のCPI_typeがEP_map typeの場合、そのCPI()はEP_mapを含む。図64に示すように、CPI()の中のCPI_typeがTU_map typeの場合、そのCPI()はTU_mapを含む。1つのAVストリームは、1つのEP_map又は1つのTU_mapを持つ。AVストリームがSESFトランスポートストリームの場合、それに対応するClipはEP_mapを持たなければならない。

図65は、CPIのシンタクスを示す図である。図65に示したCPIのシンタクスを説明するに、version_numberは、このCPI()のバージョンナンバを示す4個のキャラクタ文字である。version_numberは、ISO 646に従って、"0045"と符号化されなければならない。lengthは、このlengthフィールドの直後からCPI()の最後までCPI()のバイト数を示す32ビットの符号なし整数である。CPI_typeは、図66に示すように、1ビットのフラグであり、ClipのCPIのタイプを表す。

次に、図65に示したCPIのシンタクス内のEP_mapについて説明する。EP_mapには、2つのタイプがあり、それはビデオストリーム用のEP_mapとオーディオストリーム用のEP_mapである。EP_mapの中のEP_map_typeが、EP_mapのタイプを区別する。Clipが1つ以上のビデオストリームを含む場合、ビデオストリーム用のEP_mapが使用されなければならない。Clipがビデオストリームを含まず、1つ以上のオーディオストリームを含む場合、オーディオストリーム用のEP_mapが使用されなければならない。

ビデオストリーム用のEP_mapについて図67を参照して説明する。ビデオストリーム用のEP_mapは、stream_PID、PTS_EP_start及びRSPN_EP_startというデータを持つ。stream_PIDは、ビデオストリームを伝送するトランスポートパケットの

PIDを示す。PTS_EP_startは、ビデオストリームのシーケンスヘッダから始まるアクセスユニットのPTSを示す。RSPN_EP_startは、AVストリームの中でPTS_EP_startにより参照されるアクセスユニットの第1バイト目を含むソースパケットのアドレスを示す。

EP_map_for_one_stream_PID()と呼ばれるサブテーブルは、同じPIDを持つトランスポートパケットによって伝送されるビデオストリーム毎に作られる。Clipの中に複数のビデオストリームが存在する場合、EP_mapは複数のEP_map_for_one_stream_PID()を含んでもよい。

オーディオストリーム用のEP_mapは、stream_PID、PTS_EP_start及びRSPN_EP_startというデータを持つ。stream_PIDは、オーディオストリームを伝送するトランスポートパケットのPIDを示す。PTS_EP_startは、オーディオストリームのアクセスユニットのPTSを示す。RSPN_EP_startは、AVストリームの中でPTS_EP_startで参照されるアクセスユニットの第1バイト目を含むソースパケットのアドレスを示す。

EP_map_for_one_stream_PID()と呼ばれるサブテーブルは、同じPIDを持つトランスポートパケットによって伝送されるオーディオストリーム毎に作られる。Clipの中に複数のオーディオストリームが存在する場合、EP_mapは複数のEP_map_for_one_stream_PID()を含んでもよい。

EP_mapとSTC_Infoの関係を説明するに、1つのEP_map_for_one_stream_PID()は、STCの不連続点に関係なく1つのテーブルに作られる。RSPN_EP_startの値とSTC_Info()において定義されるRSPN_STC_startの値を比較することにより、それぞれのSTC_sequenceに属するEP_mapのデータの境界が分かる（図68を参照）。EP_mapは、同じPIDで伝送される連続したストリームの範囲に対して、1つのEP_map_for_one_stream_PIDを持たねばならない。図69に示したような場合、program#1とprogram#3は、同じビデオPIDを持つが、データ範囲が連続していないので、それぞれのプログラム毎にEP_map_for_one_stream_PIDを持たねばならない。

図70は、EP_mapのシンタクスを示す図である。図70に示したEP_mapのシンタクスを説明するに、EP_typeは、4ビットのフィールドであり、図71に示すように、EP_mapのエントリポイントタイプを示す。EP_typeは、このフィールドに続

くデータフィールドのセマンティクスを示す。Clipが1つ以上のビデオストリームを含む場合、EP_typeは0 ('video') にセットされなければならない。あるいはClipがビデオストリームを含まず、1つ以上のオーディオストリームを含む場合、EP_typeは1 ('audio') にセットされなければならない。

number_of_stream_PIDsの16ビットのフィールドは、EP_map()の中のnumber_of_stream_PIDsを変数にもつfor-loopのループ回数を示す。stream_PID(k)の16ビットのフィールドは、EP_map_for_one_stream_PID(num_EP_entries(k))によって参照されるk番目のエレメンタリストリーム（ビデオ又はオーディオストリーム）を伝送するトランスポートパケットのPIDを示す。EP_typeが0 ('video') に等しい場合、そのエレメンタリストリームはビデオストリームでなければならない。また、EP_typeが1 ('audio') に等しい場合、そのエレメンタリストリームはオーディオストリームでなければならない。

num_EP_entries(k)の16ビットのフィールドは、EP_map_for_one_stream_PID(num_EP_entries(k))によって参照されるnum_EP_entries(k)を示す。EP_map_for_one_stream_PID_Start_address(k) : この32ビットのフィールドは、EP_map()の中でEP_map_for_one_stream_PID(num_EP_entries(k))が始まる相対バイト位置を示す。この値は、EP_map()の第1バイト目からの大きさで示される。

padding_wordは、EP_map()のシンタクスに従って挿入されなければならない。XとYは、0 又は任意の正の整数でなければならない。それぞれのパディングワードは、任意の値を取ってもよい。

図72は、EP_map_for_one_stream_PIDのシンタクスを示す図である。図72に示したEP_map_for_one_stream_PIDのシンタクスを説明するに、PTS_EP_startの32ビットのフィールドのセマンティクスは、EP_map()において定義されるEP_typeにより異なる。EP_typeが0 ('video') に等しい場合、このフィールドは、ビデオストリームのシーケンスヘッダで始まるアクセスユニットの33ビット精度のPTSの上位32ビットを持つ。EP_typeが1 ('audio') に等しい場合、このフィールドは、オーディオストリームのアクセスユニットの33ビット精度のPTSの上位32ビットを持つ。

RSPN_EP_startの32ビットのフィールドのセマンティクスは、EP_map()におい

て定義されるEP_typeにより異なる。EP_typeが0 ('video') に等しい場合、このフィールドは、AVストリームの中でPTS_EP_startにより参照されるアクセスユニットのシーケンスヘッダの第1バイト目を含むソースパケットの相対アドレスを示す。あるいはEP_typeが1 ('audio') に等しい場合、このフィールドは、AVストリームの中でPTS_EP_startにより参照されるアクセスユニットのオーディオフィームの第1バイト目を含むソースパケットの相対アドレスを示す。

RSPN_EP_startは、ソースパケット番号を単位とする大きさであり、AVストリームファイルの最初のソースパケットからClipInfo()において定義されるoffset_SPNの値を初期値としてカウントされる。そのAVストリームファイルの中での絶対アドレスは、

$$\text{SPN_xxx} = \text{RSPN_xxx} - \text{offset_SPN}$$

により算出される。シンタクスのfor-loopの中でRSPN_EP_startの値は、昇順に現れなければならない。

次に、TU_mapについて、図73を参照して説明する。TU_mapは、ソースパケットのアライバルタイムクロック（到着時刻ベースの時計）に基づいて、1つの時間軸を作る。その時間軸は、TU_map_time_axisと呼ばれる。TU_map_time_axisの原点は、TU_map()の中のoffset_timeによって示される。TU_map_time_axisは、offset_timeから一定の単位に分割される。その単位を、time_unitと称する。

AVストリームの中の各々のtime_unitの中で、最初の完全な形のソースパケットのAVストリームファイル上のアドレスが、TU_mapにストアされる。これらのアドレスを、RSPN_time_unit_startと称する。TU_map_time_axis上において、k (k>=0) 番目のtime_unitが始まる時刻は、TU_start_time(k)と呼ばれる。この値は次式に基づいて算出される。

$$\text{TU_start_time}(k) = \text{offset_time} + k * \text{time_unit_size}$$

TU_start_time(k)は、45 kHzの精度を持つ。

図74は、TU_mapのシンタクスを示す図である。図74に示したTU_mapのシンタクスを説明するに、offset_timeの32ビット長のフィールドは、TU_map_time_axisに対するオフセットタイムを与える。この値は、Clipの中の最初のtime_unitに対するオフセット時刻を示す。offset_timeは、27 MHz精度のアライバルタ

イムクロックから導き出される 4 5 kHzクロックを単位とする大きさである。AVストリームが新しいClipとして記録される場合、offset_timeは 0 にセットされなければならない。

time_unit_sizeの 3 2 ビットフィールドは、time_unitの大きさを与えるものであり、それは 2 7 MHz精度のアライバルタイムクロックから導き出される 4 5 kHzクロックを単位とする大きさである。time_unit_sizeは、1 秒以下 (time_unit_size<=4 5 0 0 0) にすることがよい。number_of_time_unit_entriesの 3 2 ビットフィールドは、TU_map()の中にストアされているtime_unitのエントリ数を示す。

RSPN_time_unit_startの 3 2 ビットフィールドは、AVストリームの中でそれぞれのtime_unitが開始する場所の相対アドレスを示す。RSPN_time_unit_startは、ソースパケット番号を単位とする大きさであり、AV streamファイルの最初のソースパケットからClipInfo()において定義されるoffset_SPNの値を初期値としてカウントされる。そのAV streamファイルの中での絶対アドレスは、

$$\text{SPN}_{xxx} = \text{RSPN}_{xxx} - \text{offset_SPN}$$

により算出される。シンタクスのfor-loopの中でRSPN_time_unit_startの値は、昇順に現れなければならない。(k+1)番目のtime_unitの中にソースパケットが何もない場合、(k+1)番目のRSPN_time_unit_startは、k番目のRSPN_time_unit_startと等しくなければならない。

図 4 5 に示したzzzzz.clipのシンタクス内のClipMarkについて説明する。ClipMarkは、クリップについてのマーク情報であり、ClipMarkの中にストアされる。このマークは、記録器（記録再生装置 1）によってセットされるものであり、ユーザによってセットされるものではない。

図 7 5 は、ClipMarkのシンタクスを示す図である。図 7 5 に示したClipMarkのシンタクスを説明するに、version_numberは、このClipMark()のバージョンナンバーを示す 4 個のキャラクタ文字である。version_numberは、ISO 646に従って、“0045”と符号化されなければならない。

lengthは、このlengthフィールドの直後からClipMark()の最後までClipMark()のバイト数を示す 3 2 ビットの符号なし整数である。number_of_clip_marksは、ClipMarkの中にストアされているマークの個数を示す 1 6 ビットの符号なし整数。

number_of_clip_marksは、0であってもよい。mark_typeは、マークのタイプを示す8ビットのフィールドであり、図76に示すテーブルに従って符号化される。

mark_time_stampは、32ビットフィールドであり、マークが指定されたポイントを示すタイムスタンプをストアする。mark_time_stampのセマンティクスは、図77に示すように、Playlist()の中のCPI_typeにより異なる。

STC_sequence_idは、CPI()の中のCPI_typeがEP_map typeを示す場合、この8ビットのフィールドは、mark_time_stampが置かれているところのSTC連続区間のSTC_sequence_idを示す。CPI()の中のCPI_typeがTU_map typeを示す場合、この8ビットのフィールドは何も意味を持たず、0にセットされる。character_setの8ビットのフィールドは、mark_nameフィールドに符号化されているキャラクタ文字の符号化方法を示す。その符号化方法は、図19に示される値に対応する。

name_lengthの8ビットフィールドは、mark_nameフィールドの中に示されるマーク名のバイト長を示す。mark_nameのフィールドは、マークの名称を示す。このフィールドの中の左からname_length数のバイト数が、有効なキャラクタ文字であり、それはマークの名称を示す。mark_nameフィールドの中で、それら有効なキャラクタ文字の後の値は、どんな値が入っていてもよい。

ref_thumbnail_indexのフィールドは、マークに付加されるサムネイル画像の情報を示す。ref_thumbnail_indexフィールドが、0xFFFFでない値の場合、そのマークにはサムネイル画像が付加されており、そのサムネイル画像は、mark.thmbファイルの中にストアされている。その画像は、mark.thmbファイルの中でref_thumbnail_indexの値を用いて参照される。ref_thumbnail_indexフィールドが、0xFFFFである場合、そのマークにはサムネイル画像が付加されていない。

図78は、図75に代わるClipMarkの他のシンタクスを示す図であり、図79は、その場合における、図76に代わるmark_typeのテーブルの例を示す。reserved_for_maker_IDは、mark_typeが、0xC0から0xFFの値を示すときに、そのmark_typeを定義しているメーカーのメーカーIDを示す16ビットのフィールドである。メーカーIDは、DVRフォーマットライセンサが指定する。mark_entry()は、マーク点に指定されたポイントを示す情報であり、そのシンタクスの詳細は後述する。representative_picture_entry()は、mark_entry()によって示されるマークを代

表する画像のポイントを示す情報であり、そのシンタクスの詳細は後述する。

ClipMarkは、ユーザがAVストリームを再生するときに、その内容を視覚的に検索できるようにするために用いられる。DVRプレーヤは、GUI（グラフィカルユーザインタフェース）を使用して、ClipMarkの情報をユーザに提示する。ClipMarkの情報を視覚的に表示するためには、`mark_entry()`が示すピクチャよりもむしろ`representative_picture_entry()`が示すピクチャを示した方がよい。

図80に、`mark_entry()`と`representative_picture_entry()`の例を示す。例えば、あるプログラムが開始してから、しばらくした後（数秒後）、そのプログラムの番組名（タイトル）が表示されるとする。ClipMarkを作るときは、`mark_entry()`は、そのプログラムの開始ポイントに置き、`representative_picture_entry()`は、そのプログラムの番組名（タイトル）が表示されるポイントに置くようにしてもよい。

DVRプレーヤは、`representative_picture_entry`の画像をGUIに表示し、ユーザがその画像を指定すると、DVRプレーヤは、`mark_entry`の置かれたポイントから再生を開始する。

`mark_entry()`及び`representative_picture_entry()`のシンタクスを、図81に示す。

`mark_time_stamp`は、32ビットフィールドであり、`mark_entry()`の場合はマークが指定されたポイントを示すタイムスタンプをストアし、また`representative_picture_entry()`の場合、`mark_entry()`によって示されるマークを代表する画像のポイントを示すタイムスタンプをストアする。

次に、ClipMarkを指定するために、PTSによるタイムスタンプベースの情報を使用するのではなく、アドレスベースの情報を使用する場合の`mark_entry()`と`representative_picture_entry()`のシンタクスの例を図82に示す。

`RSPN_ref_EP_start`は、`mark_entry()`の場合、AVストリームの中でマーク点のピクチャをデコードするためのストリームのエントリポイントを示すソースパケットの相対アドレスを示す。また、`representative_picture_entry()`の場合、`mark_entry()`によって示されるマークを代表するピクチャをデコードするためのストリームのエントリポイントを示すソースパケットの相対アドレスを示す。`RSPN_r`

ef_EP_startの値は、EP_mapの中にRSPN_EP_startとしてストアされていなければならない。かつ、そのRSPN_EP_startに対応するPTS_EP_startの値は、EP_mapの中で、マーク点のピクチャのPTSより過去で最も近い値でなければならない。

offset_num_picturesは、32ビットのフィールドであり、RSPN_ref_EP_startにより参照されるピクチャから表示順序でマーク点で示されるピクチャまでのオフセットのピクチャ数を示す。この数は、0からカウントされる。図83の例の場合、offset_num_picturesは6となる。

次に、ClipMarkを指定するために、アドレススペースの情報を使用する場合のmark_entry()とrepresentative_picture_entry()のシンタクスの別の例を図84に示す。

RSPN_mark_pointは、mark_entry()の場合、AVストリームの中で、そのマークが参照するアクセスユニットの第1バイト目を含むソースパケットの相対アドレスを示す。また、representative_picture_entry()の場合、mark_entry()によって示されるマークを代表する符号化ピクチャの第1バイト目を含むソースパケットの相対アドレスを示す。

RSPN_mark_pointは、ソースパケット番号を単位とする大きさであり、AVストリームファイルの最初のソースパケットからClip Information fileにおいて定義されるoffset_SPNの値を初期値としてカウントされる。

図85を用いて、ClipMarkとEP_mapの関係を説明する。この例の場合、EP_mapが、エントリポイントのアドレスとしてI0、I1、Inを指定しており、これらのアドレスからシーケンスヘッダに続くIピクチャが開始しているとする。ClipMarkが、あるマークのアドレスとして、M1を指定しているとき、そのソースパケットから開始しているピクチャをデコードできるためには、M1のアドレスより前で最も近いエントリポイントであるI1からデータを読み出し開始すればよい。

MakersPrivateDataについては、図22を参照して既に説明したので、その説明は省略する。

次に、サムネイルインフォメーション (Thumbnail Information) について説明する。サムネイル画像は、menu.thmbファイル又はmark.thmbファイルにストアされる。これらのファイルは同じシンタクス構造であり、ただ1つのThumbnail()を

持つ。menu.thmbファイルは、メニューサムネイル画像、すなわちVolumeを代表する画像、及びそれぞれのPlayListを代表する画像をストアする。全てのメニューサムネイルは、ただ1つのmenu.thmbファイルにストアされる。

mark.thmbファイルは、マークサムネイル画像、すなわちマーク点を表すピクチャをストアする。全てのPlayList及びClipに対する全てのマークサムネイルは、ただ1つのmark.thmbファイルにストアされる。サムネイルは頻繁に追加、削除されるので、追加操作と部分削除の操作は容易に高速に実行できなければならない。この理由のため、Thumbnail()はブロック構造を有する。画像のデータは幾つかの部分に分割され、各部分は1つのtn_blockに格納される。1つの画像データは連続したtn_blockに格納される。tn_blockの列には、使用されていないtn_blockが存在してもよい。1つのサムネイル画像のバイト長は可変である。

図86は、menu.thmbとmark.thmbのシンタクスを示す図であり、図87は、図86に示したmenu.thmbとmark.thmbのシンタクス内のThumbnailのシンタクスを示す図である。図87に示したThumbnailのシンタクスについて説明するに、version_numberは、このThumbnail()のバージョンナンバを示す4個のキャラクタ文字である。version_numberは、ISO 646に従って、"0045"と符号化されなければならない。

lengthは、このlengthフィールドの直後からThumbnail()の最後までのMakersPrivateData()のバイト数を示す32ビットの符号なし整数である。tn_blocks_start_addressは、Thumbnail()の先頭のバイトからの相対バイト数を単位として、最初のtn_blockの先頭バイトアドレスを示す32ビットの符号なし整数である。相対バイト数は0からカウントされる。number_of_thumbnailsは、Thumbnail()の中に含まれているサムネイル画像のエントリ数を与える16ビットの符号なし整数である。

tn_block_sizeは、1024バイトを単位として、1つのtn_blockの大きさを与える16ビットの符号なし整数である。例えば、tn_block_size=1ならば、それは1つのtn_blockの大きさが1024バイトであることを示す。number_of_tn_blocksは、このThumbnail()中のtn_blockのエントリ数を表す16ビットの符号なし整数である。thumbnail_indexは、このthumbnail_indexフィールドから始まる

forループ1回分のサムネイル情報で表されるサムネイル画像のインデックス番号を表す16ビットの符号なし整数である。thumbnail_indexとして、0xFFFFという値を使用してはならない。thumbnail_indexはUIAppInfoVolume()、UIAppInfoPlaylist()、PlaylistMark()及びClipMark()の中のref_thumbnail_indexによって参照される。

thumbnail_picture_formatは、サムネイル画像のピクチャフォーマットを表す8ビットの符号なし整数で、図88に示すような値をとる。表中のDCFとPNGは"menu.thmb"内でのみ許される。マークサムネイルは、値"0x00" (MPEG-2 Video I-picture) をとらなければならない。

picture_data_sizeは、サムネイル画像のバイト長をバイト単位で示す32ビットの符号なし整数である。start_tn_block_numberは、サムネイル画像のデータが始まるtn_blockのtn_block番号を表す16ビットの符号なし整数である。サムネイル画像データの先頭は、tb_blockの先頭と一致していなければならない。tn_block番号は、0から始まり、tn_blockのfor-loop中の変数kの値に関係する。

x_picture_lengthは、サムネイル画像のフレーム画枠の水平方向のピクセル数を表す16ビットの符号なし整数である。y_picture_lengthは、サムネイル画像のフレーム画枠の垂直方向のピクセル数を表す16ビットの符号なし整数である。tn_blockは、サムネイル画像がストアされる領域である。Thumbnail()の中の全てのtn_blockは、同じサイズ(固定長)であり、その大きさはtn_block_sizeによって定義される。

図89A、図89Bは、サムネイル画像データがどのようにtn_blockに格納されるかを模式的に表した図である。図89A、図89Bのように、各サムネイル画像データはtn_blockの先頭から始まり、1tn_blockを超える大きさの場合は、連続する次のtn_blockを使用してストアされる。このようにすることにより、可変長であるピクチャデータが、固定長のデータとして管理することが可能となり、削除といった編集に対して簡便な処理により対応することができるようになる。

次に、AVストリームファイルについて説明する。AVストリームファイルは、"M2TS"ディレクトリ(図14)にストアされる。AVストリームファイルには、2つのタイプがあり、それらは、Clip AVストリームとBridge-Clip AVストリームファ

イルである。両方のAVストリーム共に、これ以降で定義されるDVR MPEG-2トランスポートストリームファイルの構造でなければならない。

まず、DVR MPEG-2トランスポートストリームについて説明する。DVR MPEG-2トランスポートストリームの構造は、図90に示すようになっている。AVストリームファイルは、DVR MPEG-2トランスポートストリームの構造を持つ。DVR MPEG-2トランスポートストリームは、整数個のAligned unitから構成される。Aligned unitの大きさは、6144バイト（2048*3バイト）である。Aligned unitは、ソースパケットの第1バイト目から始まる。ソースパケットは、192バイト長である。1つのソースパケットは、TP_extra_headerとトランスポートパケットから成る。TP_extra_headerは、4バイト長であり、またトランスポートパケットは、188バイト長である。

1つのAligned unitは、32個のソースパケットから成る。DVR MPEG-2トランスポートストリームの中の最後のAligned unitも、また32個のソースパケットから成る。よって、DVR MPEG-2トランスポートストリームは、Aligned unitの境界で終端する。ディスクに記録される入力トランスポートストリームのトランスポートパケットの数が32の倍数でないとき、ヌルパケット（PID=0x1FFFのトランスポートパケット）を持ったソースパケットを最後のAligned unitに使用しなければならない。ファイルシステムは、DVR MPEG-2トランスポートストリームに余分な情報を付加してはならない。

図91に、DVR MPEG-2トランスポートストリームのレコーダモデルを示す。図91に示したレコーダは、レコーディングプロセスを規定するための概念上のモデルである。DVR MPEG-2トランスポートストリームは、このモデルに従う。

MPEG-2トランスポートストリームの入力タイミングについて説明する。入力MPEG-2トランスポートストリームは、フルトランスポートストリーム又はパーシャルトランスポートストリームである。入力されるMPEG-2トランスポートストリームは、ISO/IEC13818-1又はISO/IEC13818-9に従っていなければならない。MPEG-2トランスポートストリームのi番目のバイトは、T-STD（ISO/IEC 13818-1で規定されるTransport stream system target decoder）51とソースパケットタイザ（source packetizer）54へ、時刻t(i)に同時に入力される。Rpkは、トランスポート

トケットの入力レートの瞬時的な最大値である。

2 7 MHz PLL 5 2 は、2 7 MHzクロックの周波数を発生する。2 7 MHzクロックの周波数は、MPEG-2トランスポートストリームのPCR (Program Clock Reference) の値にロックされる。アライバルタイムクロックカウンタ (arrival time clock counter) 5 3 は、2 7 MHzの周波数のパルスをカウントするバイナリカウンタである。arrival_time_clock(i)は、時刻t(i)におけるarrival time clock counter 5 3 のカウント値である。

source packetizer 5 4 は、全てのトランスポートケットにTP_extra_headerを付加し、ソースケットを作る。arrival_time_stampは、トランスポートケットの第1バイト目がT-STD 5 1 とソースパケッタイザ 5 4 の両方へ到着する時刻を表す。arrival_time_stamp(k)は、次式で示されるようにarrival_time_clock(k)のサンプル値であり、ここで、kはトランスポートケットの第1バイト目を示す。

$$\text{arrival_time_stamp}(k) = \text{arrival_time_clock}(k) \% 230$$

2つの連続して入力されるトランスポートケットの時間間隔が、 $230 / 27000000$ 秒 (約40秒) 以上になる場合、その2つのトランスポートケットのarrival_time_stampの差分は、 $230 / 27000000$ 秒になるようにセットされるべきである。レコーダは、そのようになる場合に備えてある。

スムージングバッファ (smoothing buffer) 5 5 は、入力トランスポートストリームのビットレートをスムージングする。スムージングバッファ 5 5 は、オーバーフローしてはならない。Rmaxは、スムージングバッファ 5 5 が空でないときのスムージングバッファ 5 5 からのソースケットの出力ビットレートである。スムージングバッファ 5 5 が空であるとき、スムージングバッファ 5 5 からの出力ビットレートは0である。

次に、DVR MPEG-2トランスポートストリームのレコーダモデルのパラメータについて説明する。Rmaxという値は、AVストリームファイルに対応するClipInfo()において定義されるTS_recording_rateによって与えられる。この値は、次式により算出される。

$$R_{\max} = \text{TS_recording_rate} * 192/188$$

TS_recording_rateの値は、bytes/secondを単位とする大きさである。

入力トランスポートストリームがSESFトランスポートストリームの場合、Rpkは、AVストリームファイルに対応するClipInfo()において定義されるTS_recording_rateに等しくなければならない。入力トランスポートストリームがSESFトランスポートストリームでない場合、この値はMPEG-2 transport streamのデスクリプタ、例えばmaximum_bitrate_descriptorやpartial_transport_stream_descriptorなどにおいて定義される値を参照してもよい。

入力トランスポートストリームがSESFトランスポートストリームの場合、スムージングバッファ55の大きさ (smoothing buffer size) は0である。入力トランスポートストリームがSESFトランスポートストリームでない場合、スムージングバッファ55の大きさはMPEG-2 transport streamのデスクリプタ、例えばsmoothing_buffer_descriptor、short_smoothing_buffer_descriptor、partial_transport_stream_descriptorなどにおいて定義される値を参照してもよい。

記録機（レコーダ）及び再生機（プレーヤ）は、十分なサイズのバッファを用意しなければならない。デフォルトのバッファサイズは、1536バイトである。

次に、DVR MPEG-2トランスポートストリームのプレーヤモデルについて説明する。図92は、DVR MPEG-2トランスポートストリームのプレーヤモデルを示す図である。これは、再生プロセスを規定するための概念上のモデルである。DVR MPEG-2トランスポートストリームは、このモデルに従う。

27MHz X-tal（クリスタル発振器）61は、27MHzの周波数を発生する。27MHz周波数の誤差範囲は、 ± 30 ppm (27000000 \pm 810 Hz) でなければならない。arrival time clock counter62は、27MHzの周波数のパルスをカウントするバイナリカウンタである。arrival_time_clock(i)は、時刻t(i)におけるarrival time clock counter62のカウント値である。

smoothing buffer64において、Rmaxは、スムージングバッファ64がフルでないときのスムージングバッファ64へのソースパケットの入力ビットレートである。スムージングバッファ64がフルであるとき、スムージングバッファ64への入力ビットレートは0である。

MPEG-2トランスポートストリームの出力タイミングを説明するに、現在のソースパケットのarrival_time_stampがarrival_time_clock(i)のLSB 30ビットの値と等しいとき、そのソースパケットのトランスポートパケットは、スムージングバッファ64から引き抜かれる。Rpkは、トランスポートパケットレートの瞬時的な最大値である。スムージングバッファ64は、アンダーフローしてはならない。

DVR MPEG-2トランスポートストリームのプレーヤモデルのパラメータについては、上述したDVR MPEG-2トランスポートストリームのレコーダモデルのパラメータと同一である。

図93は、Source packetのシンタクスを示す図である。transport_packet()は、ISO/IEC 13818-1で規定されるMPEG-2トランスポートパケットである。図93に示したSource packetのシンタクス内のTP_extra_headerのシンタクスを図94に示す。図94に示したTP_extra_headerのシンタクスについて説明するに、copy_permission_indicatorは、トランスポートパケットのペイロードのコピー制限を表す整数である。コピー制限は、copy free、no more copy、copy once又はcopy prohibitedとすることができる。図95は、copy_permission_indicatorの値と、それらによって指定されるモードの関係を示す。

copy_permission_indicatorは、全てのトランスポートパケットに付加される。IEEE1394デジタルインタフェースを使用して入力トランスポートストリームを記録する場合、copy_permission_indicatorの値は、IEEE1394 isochronous packet headerの中のEMI (Encryption Mode Indicator) の値に関連付けてもよい。IEEE1394デジタルインタフェースを使用しないで入力トランスポートストリームを記録する場合、copy_permission_indicatorの値は、トランスポートパケットの中に埋め込まれたCCIの値に関連付けてもよい。アナログ信号入力をセルフエンコードする場合、copy_permission_indicatorの値は、アナログ信号のCGMS-Aの値に関連付けてもよい。

arrival_time_stampは、次式

$$\text{arrival_time_stamp}(k) = \text{arrival_time_clock}(k) \% 230$$

において、arrival_time_stampによって指定される値を持つ整数値である。

Clip AVストリームの定義をするに、Clip AVストリームは、上述したような定

義がされるDVR MPEG-2トランスポートストリームの構造を持たねばならない。arrival_time_clock(i)は、Clip AVストリームの中で連続して増加しなければならない。Clip AVストリームの中にシステムタイムベース（STCベース）の不連続点が存在したとしても、そのClip AVストリームのarrival_time_clock(i)は、連続して増加しなければならない。

Clip AVストリームの中の開始と終了の間のarrival_time_clock(i)の差分の最大値は、26時間でなければならない。この制限は、MPEG-2トランスポートストリームの中にシステムタイムベース（STCベース）の不連続点が存在しない場合に、Clip AVストリームの中で同じ値のPTS（Presentation Time Stamp）が決して現れないことを保証する。MPEG2システムズ規格は、PTSのラップアラウンド周期を233/90000秒（約26.5時間）と規定している。

Bridge-Clip AVストリームの定義をするに、Bridge-Clip AVストリームは、上述したような定義がされるDVR MPEG-2トランスポートストリームの構造を持たねばならない。Bridge-Clip AVストリームは、1つのアライバルタイムベースの不連続点を含まなければならない。アライバルタイムベースの不連続点の前後のトランスポートストリームは、後述する符号化の制限に従わなければならない、かつ後述するDVR-STDに従わなければならない。

本実施の形態においては、編集におけるPlayItem間のビデオとオーディオのシームレス接続をサポートする。PlayItem間をシームレス接続にすることは、プレーヤ/レコーダに”データの連続供給”と”シームレスな復号処理”を保証する。”データの連続供給”とは、ファイルシステムが、デコーダにバッファのアンダーフローを起こさせることのないように必要なビットレートでデータを供給することを保証できることである。データのリアルタイム性を保証して、データをディスクから読み出すことができるように、データが十分な大きさの連続したブロック単位でストアされるようにする。

”シームレスな復号処理”とは、プレーヤが、デコーダの再生出力にポーズやギャップを起こさせることなく、ディスクに記録されたオーディオビデオデータを表示できることである。

シームレス接続されているPlayItemが参照するAVストリームについて説明する。

先行するPlayItemと現在のPlayItemの接続が、シームレス表示できるように保証されているかどうかは、現在のPlayItemにおいて定義されているConnection_Conditionフィールドから判断することができる。PlayItem間のシームレス接続は、Bridge-Clipを使用する方法と使用しない方法がある。

図9 6は、Bridge-Clipを使用する場合の先行するPlayItemと現在のPlayItemの関係を示している。図9 6においては、プレーヤが読み出すストリームデータが、影を付けて示されている。図9 6に示したTS1は、Clip1 (Clip AVストリーム) の影を付けられたストリームデータとBridge-ClipのRSPN_arrival_time_discontinuityより前の影を付けられたストリームデータから成る。

TS1のClip1の影を付けられたストリームデータは、先行するPlayItemのIN_time (図9 6においてIN_time1で図示されている) に対応するプレゼンテーションユニットを復号するために必要なストリームのアドレスから、RSPN_exit_from_previous_Clipで参照されるソースパケットまでのストリームデータである。TS1に含まれるBridge-ClipのRSPN_arrival_time_discontinuityより前の影を付けられたストリームデータは、Bridge-Clipの最初のソースパケットから、RSPN_arrival_time_discontinuityで参照されるソースパケットの直前のソースパケットまでのストリームデータである。

また、図9 6におけるTS2は、Clip2 (Clip AVストリーム) の影を付けられたストリームデータとBridge-ClipのRSPN_arrival_time_discontinuity以後の影を付けられたストリームデータから成る。TS2に含まれるBridge-ClipのRSPN_arrival_time_discontinuity以後の影を付けられたストリームデータは、RSPN_arrival_time_discontinuityで参照されるソースパケットから、Bridge-Clipの最後のソースパケットまでのストリームデータである。TS2のClip2の影を付けられたストリームデータは、RSPN_enter_to_current_Clipで参照されるソースパケットから、現在のPlayItemのOUT_time (図9 6においてOUT_time2で図示されている) に対応するプレゼンテーションユニットを復号するために必要なストリームのアドレスまでのストリームデータである。

図9 7は、Bridge-Clipを使用しない場合の先行するPlayItemと現在のPlayItemの関係を示している。この場合、プレーヤが読み出すストリームデータは、影を

付けて示されている。図 9 7 における TS1 は、Clip1 (Clip AV ストリーム) の影を付けられたストリームデータから成る。TS1 の Clip1 の影を付けられたストリームデータは、先行する PlayItem の IN_time (図 9 7 において IN_time1 で図示されている) に対応するプレゼンテーションユニットを復号するために必要なストリームのアドレスから始まり、Clip1 の最後のソースパケットまでのデータである。また、図 9 7 における TS2 は、Clip2 (Clip AV ストリーム) の影を付けられたストリームデータから成る。

TS2 の Clip2 の影を付けられたストリームデータは、Clip2 の最初のソースパケットから始まり、現在の PlayItem の OUT_time (図 9 7 において OUT_time2 で図示されている) に対応するプレゼンテーションユニットを復号するために必要なストリームのアドレスまでのストリームデータである。

図 9 6 と図 9 7 において、TS1 と TS2 は、ソースパケットの連続したストリームである。次に、TS1 と TS2 のストリーム規定と、それらの間の接続条件について考える。まず、シームレス接続のための符号化制限について考える。トランスポートストリームの符号化構造の制限として、まず、TS1 と TS2 の中に含まれるプログラムの数は、1 でなければならない。TS1 と TS2 の中に含まれるビデオストリームの数は、1 でなければならない。TS1 と TS2 の中に含まれるオーディオストリームの数は、2 以下でなければならない。TS1 と TS2 の中に含まれるオーディオストリームの数は、等しくなければならない。TS1 及び／又は TS2 の中に、上記以外のエレメンタリストリーム又はプライベートストリームが含まれていてもよい。

ビデオビットストリームの制限について説明する。図 9 8 は、ピクチャの表示順序で示すシームレス接続の例を示す図である。接続点においてビデオストリームをシームレスに表示できるためには、OUT_time1 (Clip1 の OUT_time) の後と IN_time2 (Clip2 の IN_time) の前に表示される不必要なピクチャは、接続点付近の Clip の部分的なストリームを再エンコードするプロセスにより、除去されなければならない。

図 9 8 に示したような場合において、BridgeSequence を使用してシームレス接続を実現する例を、図 9 9 に示す。RSPN_arrival_time_discontinuity より前の Bridge-Clip のビデオストリームは、図 9 8 の Clip1 の OUT_time1 に対応するピクチャ

ャまでの符号化ビデオストリームから成る。そして、そのビデオストリームは先行するClip1のビデオストリームに接続され、1つの連続でMPEG2規格に従ったエレメンタリストリームとなるように再エンコードされている。

同様に、RSPN_arrival_time_discontinuity以後のBridge-Clipのビデオストリームは、図98のClip2のIN_time2に対応するピクチャ以後の符号化ビデオストリームから成る。そして、そのビデオストリームは、正しくデコード開始することができて、これに続くClip2のビデオストリームに接続され、1つの連続でMPEG2規格に従ったエレメンタリストリームとなるように再エンコードされている。Bridge-Clipを作るためには、一般に、数枚のピクチャは再エンコードしなければならない、それ以外のピクチャはオリジナルのClipからコピーすることができる。

図98に示した例の場合にBridgeSequenceを使用しないでシームレス接続を実現する例を図100に示す。Clip1のビデオストリームは、図98のOUT_time1に対応するピクチャまでの符号化ビデオストリームから成り、それは、1つの連続でMPEG2規格に従ったエレメンタリストリームとなるように再エンコードされている。同様に、Clip2のビデオストリームは、図98のClip2のIN_time2に対応するピクチャ以後の符号化ビデオストリームから成り、それは、1つの連続でMPEG2規格に従ったエレメンタリストリームとなるように再エンコードされている。

ビデオストリームの符号化制限について説明するに、まず、TS1とTS2のビデオストリームのフレームレートは、等しくなければならない。TS1のビデオストリームは、sequence_end_codeで終端しなければならない。TS2のビデオストリームは、Sequence Header、GOP Header、そしてI-ピクチャで開始しなければならない。TS2のビデオストリームは、クローズドGOPで開始しなければならない。

ビットストリームの中で定義されるビデオプレゼンテーションユニット（フレーム又はフィールド）は、接続点を挟んで連続でなければならない。接続点において、フレーム又はフィールドのギャップがあってはならない。接続点において、トップ?ボトムのフィールドシーケンスは連続でなければならない。3-2プルダウンを使用するエンコードの場合は、"top_field_first"及び"repeat_first_field"フラグを書き換える必要があるかもしれない、又はフィールドギャップの発生を防ぐために局所的に再エンコードするようにしてもよい。

オーディオビットストリームの符号化制限について説明するに、TS1とTS2のオーディオのサンプリング周波数は、同じでなければならない。TS1とTS2のオーディオの符号化方法（例えば、MPEG1レイヤ2、AC-3、SESF LPCM、AAC）は、同じでなければならない。

次に、MPEG-2トランスポートストリームの符号化制限について説明するに、TS1のオーディオストリームの最後のオーディオフレームは、TS1の最後の表示ピクチャの表示終了時に等しい表示時刻を持つオーディオサンプルを含んでいなければならない。TS2のオーディオストリームの最初のオーディオフレームは、TS2の最初の表示ピクチャの表示開始時に等しい表示時刻を持つオーディオサンプルを含んでいなければならない。

接続点において、オーディオプレゼンテーションユニットのシーケンスにギャップがあってはならない。図101に示すように、2オーディオフレーム区間未満のオーディオプレゼンテーションユニットの長さで定義されるオーバーラップがあってもよい。TS2のエレメンタリストリームを伝送する最初のパケットは、ビデオパケットでなければならない。接続点におけるトランスポートストリームは、後述するDVR-STDに従わなくてはならない。

Clip及びBridge-Clipの制限について説明するに、TS1とTS2は、それぞれの中にアライバルタイムベースの不連続点を含んではならない。

以下の制限は、Bridge-Clipを使用する場合にのみ適用される。TS1の最後のソースパケットとTS2の最初のソースパケットの接続点においてのみ、Bridge-Clip AVストリームは、ただ1つのアライバルタイムベースの不連続点を持つ。ClipInfo()において定義されるRSPN_arrival_time_discontinuityが、その不連続点のアドレスを示し、それはTS2の最初のソースパケットを参照するアドレスを示さなければならない。

BridgeSequenceInfo()において定義されるRSPN_exit_from_previous_Clipによって参照されるソースパケットは、Clip1の中のどのソースパケットでもよい。それは、Aligned unitの境界である必要はない。BridgeSequenceInfo()において定義されるRSPN_enter_to_current_Clipによって参照されるソースパケットは、Clip2の中のどのソースパケットでもよい。それは、Aligned unitの境界である必要

はない。

PlayItemの制限について説明するに、先行するPlayItemのOUT_time (図 9 6、図 9 7 において示されるOUT_time1) は、TS1の最後のビデオプレゼンテーションユニットの表示終了時刻を示さなければならない。現在のPlayItemのIN_time (図 9 6、図 9 7 において示されるIN_time2) は、TS2の最初のビデオプレゼンテーションユニットの表示開始時刻を示さなければならない。

Bridge-Clipを使用する場合のデータアロケーションの制限について、図 1 0 2 を参照して説明するに、シームレス接続は、ファイルシステムによってデータの連続供給が保証されるように作られなければならない。これは、Clip1 (Clip AVストリームファイル) とClip2 (Clip AVストリームファイル) に接続されるBridge-Clip AVストリームを、データアロケーション規定を満たすように配置することによって行われなければならない。

RSPN_exit_from_previous_Clip以前のClip1 (Clip AVストリームファイル) のストリーム部分が、ハーフフラグメント以上の連続領域に配置されているように、RSPN_exit_from_previous_Clipが選択されなければならない。Bridge-Clip AVストリームのデータ長は、ハーフフラグメント以上の連続領域に配置されるように、選択されなければならない。RSPN_enter_to_current_Clip以後のClip2 (Clip AVストリームファイル) のストリーム部分が、ハーフフラグメント以上の連続領域に配置されているように、RSPN_enter_to_current_Clipが選択されなければならない。

Bridge-Clipを使用しないでシームレス接続する場合のデータアロケーションの制限について、図 1 0 3 を参照して説明するに、シームレス接続は、ファイルシステムによってデータの連続供給が保証されるように作られなければならない。これは、Clip1 (Clip AVストリームファイル) の最後の部分とClip2 (Clip AVストリームファイル) の最初の部分を、データアロケーション規定を満たすように配置することによって行われなければならない。

Clip1 (Clip AVストリームファイル) の最後のストリーム部分が、ハーフフラグメント以上の連続領域に配置されていなければならない。Clip2 (Clip AVストリームファイル) の最初のストリーム部分が、ハーフフラグメント以上の連続領

域に配置されていなければならない。

次に、DVR-STDについて説明する。DVR-STDは、DVR MPEG-2トランスポートストリームの生成及び検証の際におけるデコード処理をモデル化するための概念モデルである。また、DVR-STDは、上述したシームレス接続された2つのPlayItemによって参照されるAVストリームの生成及び検証の際におけるデコード処理をモデル化するための概念モデルでもある。

DVR-STDモデルを図104に示す。図104に示したモデルには、DVR MPEG-2トランスポートストリームプレーヤモデルが構成要素として含まれている。n、TBn、MBn、EBn、TBsys、Bsys、Rxn、Rbxn、Rxsys、Dn、Dsys、On及びPn(k)の表記方法は、ISO/IEC13818-1のT-STDに定義されているものと同じである。すなわち、次の通りである。nは、エレメンタリストリームのインデックス番号である。TBnは、エレメンタリストリームnのトランスポートバッファである。

MBnは、エレメンタリストリームnの多重バッファである。ビデオストリームについてのみ存在する。EBnは、エレメンタリストリームnのエレメンタリストリームバッファである。ビデオストリームについてのみ存在する。TBsysは、復号中のプログラムのシステム情報のための入力バッファである。Bsysは、復号中のプログラムのシステム情報のためのシステムターゲットデコーダ内のメインバッファである。Rxnは、データがTBnから取り除かれる伝送レートである。Rbxnは、PESパケットペイロードがMBnから取り除かれる伝送レートである。ビデオストリームについてのみ存在する。

Rxsysは、データがTBsysから取り除かれる伝送レートである。Dnは、エレメンタリストリームnのデコーダである。Dsysは、復号中のプログラムのシステム情報に関するデコーダである。Onは、ビデオストリームnのre-ordering bufferである。Pn(k)は、エレメンタリストリームnのk番目のプレゼンテーションユニットである。

DVR-STDのデコーディングプロセスについて説明する。単一のDVR MPEG-2トランスポートストリームを再生している間は、トランスポートパケットをTB1、TBn又はTBsysのバッファへ入力するタイミングは、ソースパケットのarrival_time_stampにより決定される。TB1、MB1、EB1、TBn、Bn、TBsys及びBsysのバッファリング動作の規定は、ISO/IEC 13818-1に規定されているT-STDと同じである。復号動

作と表示動作の規定もまた、ISO/IEC 13818-1に規定されているT-STDと同じである。

シームレス接続されたPlayItemを再生している間のデコーディングプロセスについて説明する。ここでは、シームレス接続されたPlayItemによって参照される2つのAVストリームの再生について説明をすることにし、以後の説明では、上述した（例えば、図96に示した）TS1とTS2の再生について説明する。TS1は、先行するストリームであり、TS2は、現在のストリームである。

図105は、あるAVストリーム（TS1）からそれにシームレスに接続された次のAVストリーム（TS2）へと移るときのトランスポートバケットの入力、復号、表示のタイミングチャートを示す。所定のAVストリーム（TS1）からそれにシームレスに接続された次のAVストリーム（TS2）へと移る間には、TS2のアライバルタイムベースの時間軸（図105においてATC2で示される）は、TS1のアライバルタイムベースの時間軸（図105においてATC1で示される）と同じでない。

また、TS2のシステムタイムベースの時間軸（図105においてSTC2で示される）は、TS1のシステムタイムベースの時間軸（図105においてSTC1で示される）と同じでない。ビデオの表示は、シームレスに連続していることが要求される。オーディオのプレゼンテーションユニットの表示時間にはオーバーラップがあってもよい。

DVR-STDへの入力タイミングについて説明する。時刻T1までの時間、すなわち、TS1の最後のビデオバケットがDVR-STDのTB1に入力終了するまでは、DVR-STDのTB1、TBn又はTBsysのバッファへの入力タイミングは、TS1のソースバケットのarrival_time_stampによって決定される。

TS1の残りのバケットは、TS_recording_rate(TS1)のビットレートでDVR-STDのTBn又はTBsysのバッファへ入力されなければならない。ここで、TS_recording_rate(TS1)は、Clip1に対応するClipInfo()において定義されるTS_recording_rateの値である。TS1の最後のバイトがバッファへ入力する時刻は、時刻T2である。したがって、時刻T1からT2までの区間では、ソースバケットのarrival_time_stampは無視される。

N1をTS1の最後のビデオバケットに続くTS1のトランスポートバケットのバイト

数とすると、時刻 T1 乃至 T2 までの時間 DT1 は、N1 バイトが TS_recording_rate(TS1) のビットレートで入力終了するために必要な時間であり、次式により算出される。

$$DT1 = T2 - T1 = N1 / TS_recording_rate(TS1)$$

時刻 T1 乃至 T2 までの間は、RXn と RXsys の値は共に、TS_recording_rate(TS1) の値に変化する。このルール以外のバッファリング動作は、T-STD と同じである。

T2 の時刻において、arrival time clock counter は、TS2 の最初のソースパケットの arrival_time_stamp の値にリセットされる。DVR-STD の TB1、TBn 又は TBsys のバッファへの入力タイミングは、TS2 のソースパケットの arrival_time_stamp によって決定される。RXn と RXsys は共に、T-STD において定義されている値に変化する。

付加的なオーディオバッファリング及びシステムデータバッファリングについて説明するに、オーディオデコーダとシステムデコーダは、時刻 T1 から T2 までの区間の入力データを処理することができるように、T-STD で定義されるバッファ量に加えて付加的なバッファ量（約 1 秒分のデータ量）が必要である。

ビデオのプレゼンテーションタイミングについて説明するに、ビデオプレゼンテーションユニットの表示は、接続点を通して、ギャップなしに連続でなければならない。ここで、STC1 は、TS1 のシステムタイムベースの時間軸（図 105 では STC1 と図示されている）とし、STC2 は、TS2 のシステムタイムベースの時間軸（図 97 では STC2 と図示されている。正確には、STC2 は、TS2 の最初の PCR が T-STD に入力した時刻から開始する。）とする。

STC1 と STC2 の間のオフセットは、次のように決定される。PTS1end は、TS1 の最後のビデオプレゼンテーションユニットに対応する STC1 上の PTS であり、PTS2start は、TS2 の最初のビデオプレゼンテーションユニットに対応する STC2 上の PTS であり、Tpp は、TS1 の最後のビデオプレゼンテーションユニットの表示期間とすると、2 つのシステムタイムベースの間のオフセット STC_delta は、次式により算出される。

$$STC_delta = PTS1end + Tpp - PTS2start$$

オーディオのプレゼンテーションのタイミングについて説明するに、接続点に

において、オーディオプレゼンテーションユニットの表示タイミングのオーバーラップがあってもよく、それは0乃至2オーディオフレーム未満である（図105に図示されている“audio overlap”を参照）。どちらのオーディオサンプルを選択するかということと、オーディオプレゼンテーションユニットの表示を接続点の後の補正されたタイムベースに再同期することは、プレーヤ側により設定されることである。

DVR-STDのシステムタイムクロックについて説明するに、時刻T5において、TS1の最後のオーディオプレゼンテーションユニットが表示される。システムタイムクロックは、時刻T2からT5の間にオーバーラップしていてもよい。この区間では、DVR-STDは、システムタイムクロックを古いタイムベースの値（STC1）と新しいタイムベースの値（STC2）の間で切り換える。STC2の値は、次式により算出される。

$$STC2 = STC1 - STC_delta$$

バッファリングの連続性について説明する。STC11video_endは、TS1の最後のビデオパケットの最後のバイトがDVR-STDのTB1へ到着するときのシステムタイムベースSTC1上のSTCの値である。STC22video_startは、TS2の最初のビデオパケットの最初のバイトがDVR-STDのTB1へ到着するときのシステムタイムベースSTC2上のSTCの値である。STC21video_endは、STC11video_endの値をシステムタイムベースSTC2上の値に換算した値である。STC21video_endは、次式により算出される。

$$STC21video_end = STC11video_end - STC_delta$$

DVR-STDに従うために、次の2つの条件を満たすことが要求される。まず、TS2の最初のビデオパケットのTB1への到着タイミングは、次に示す不等式を満たさなければならない。そして、次に示す不等式を満たさなければならない。

$$STC22video_start > STC21video_end + \Delta T1$$

この不等式が満たされるように、Clip1及び／又はClip2の部分的なストリームを再エンコード及び／又は再多重化する必要がある場合は、その必要に応じて行われる。

次に、STC1とSTC2を同じ時間軸上に換算したシステムタイムベースの時間軸上において、TS1からのビデオパケットの入力とそれに続くTS2からのビデオパケッ

トの入力は、ビデオバッファをオーバーフロー及びアンダーフローさせてはならない。

このようなシンタクス、データ構造、規則に基づくことにより、記録媒体に記録されているデータの内容、再生情報などを適切に管理することができ、もって、ユーザが再生時に適切に記録媒体に記録されているデータの内容を確認したり、所望のデータを簡便に再生できるようにすることができる。

なお、本実施の形態は、多重化ストリームとしてMPEG-2トランスポートストリームを例にして説明しているが、これに限らず、MPEG2プログラムストリームや米国のDirecTVサービス（商標）で使用されているDSSトランスポートストリームについても適用することが可能である。

次に、mark_entry()及びrepresentative_picture__entry()のシンタクスが、図81に示されるような構成である場合における、マーク点で示されるシーンの頭出し再生を行う場合の処理について、図106のフローチャートを参照して、説明する。

最初にステップS1において、記録再生装置1の制御部23は、記録媒体100から、DVRトランスポートストリームファイルのデータベースであるEP_map（図70）、STC_Info（図52）、Program_Info（図54）及びClipMark（図78）を読み出す。

ステップS2において、制御部23は、ClipMark（図78）のrepresentative_picture_entry（図81）又はref_thumbnail_indexで参照されるピクチャからサムネイルのリストを作成し、ユーザインタフェース入出力としての端子24から出力し、GUIのメニュー画面上に表示させる。この場合、ref_thumbnail_indexが有効な値を持つ場合、representative_picture_entryよりref_thumbnail_indexが優先される。

ステップS3において、ユーザが再生開始点のマーク点を指定する。これは、例えば、GUIとして表示されたメニュー画面上の中からユーザがサムネイル画像を選択することで行われる。制御部23は、この選択操作に対応して、指定されたサムネイルに対応付けられているマーク点を取得する。

ステップS4において、制御部23は、ステップS3で指定されたmark_entry

(図 8 1) の mark_Time_stamp の PTS と、 STC_sequence_id を取得する。

ステップ S 5 において、制御部 2 3 は、 STC_Info (図 5 2) から、ステップ S 4 で取得した STC_sequence_id に対応する STC 時間軸が開始するソースパケット番号を取得する。

ステップ S 6 において、制御部 2 3 は、ステップ S 5 で取得した STC 時間軸が開始するパケット番号と、ステップ S 4 で取得したマーク点の PTS から、マーク点の PTS より時間的に前で、かつ、最も近いエントリポイント (I ピクチャ) のあるソースパケット番号を取得する。

ステップ S 7 において、制御部 2 3 は、ステップ S 6 で取得したエントリポイントのあるソースパケット番号から、トランスポートストリームのデータを読み出し、AVデコーダ 2 7 に供給させる。

ステップ S 8 において、制御部 2 3 は、AVデコーダ 2 7 を制御し、ステップ S 4 で取得したマーク点の PTS のピクチャから表示を開始させる。

以上の動作を、図 1 0 7 乃至 1 0 9 を参照してさらに説明する。

いま、図 1 0 7 に示されているように、DVR トランスポートストリームファイルは、STC_sequence_id=id 0 の STC 時間軸を有し、その時間軸が開始するソースパケット番号は、シーン開始点 A のソースパケット番号より小さいものとする。そして、ソースパケット番号 B から C までの間に、CM (コマーシャル) が挿入されているものとする。

このとき、図 7 0 に示される EP_map に対応する EP_map には、図 1 0 8 に示されるように、RSPN_EP_start で示される A、B、C に対応して、それぞれの PTS が、PTS_EP_start として、PTS(A)、PTS(B)、PTS(C) として登録される。

また、図 1 0 9 に示されるように、図 7 8 の ClipMark に対応する ClipMark には、図 1 0 9 に示されるように、シーンスタート、CM スタート及び CM エンドを表すマークタイプ (図 7 9) 0 x 9 2, 0 x 9 4, 0 x 9 5 の値に対応して、mark_entry と representative_picture_entry が記録される。

mark_entry の Mark_Time_stamp としては、シーンスタート、CM スタート及び CM エンドに対応して、それぞれ PTS(a1)、PTS(b0)、PTS(c0) が登録されており、それぞれの STC_sequence_id は、いずれも id 0 とされている。

同様に、representative_picture_entryのMark_Time_stampとして、シーンスタート、CMスタート及びCMエンドに対応して、それぞれPTS(a2)、PTS(b0)、PTS(c0)が登録されており、それらはいずれもSTC_sequence_idが、id0とされている。

PTS(A)<PTS(a1)の場合、ステップS6において、パケット番号Aが取得され、ステップS7において、パケット番号Aから始まるトランスポートストリームが、AVデコーダ27に供給され、ステップS8において、PTS(a1)のピクチャから表示が開始される。

次に、図110のフローチャートを参照して、mark_entryとrepresentative_picture_entryのシンタクスが、図81に示されるような構成である場合におけるCMスキップ再生の処理について、図110のフローチャートを参照して説明する。

ステップS21において、制御部23は、EP_map(図70)、STC_Info(図52)、Program_Info(図54)及びClipMark(図78)を記録媒体100から読み出す。ステップS22において、ユーザは、ユーザインタフェース入出力としての端子24からCMスキップ再生を指定する。

ステップS23において、制御部23は、マークタイプ(図79)がCM開始点(0x94)であるマーク情報のPTSと、CM終了点(0x95)であるマーク情報のPTS、並びに対応するSTC_sequence_idを取得する(図81)。

ステップS24において、制御部23は、STC_Info(図52)からCM開始点と終了点の、STC_sequence_idに対応するSTC時間軸が開始するソースパケット番号を取得する。

ステップS25において、制御部23は、記録媒体100からトランスポートストリームを読み出させ、それをAVデコーダ27に供給し、デコードを開始させる。

ステップS26において、制御部23は、現在の表示画像がCM開始点のPTSの画像か否かを調べる。現在の表示画像がCM開始点のPTSの画像でない場合には、ステップS27に進み、制御部23は、画像の表示が継続される。その後、処理はステップS25に戻り、それ以降の処理が繰り返し実行される。

ステップS26において、現在の表示画像がCM開始点のPTSの画像であると判定された場合、ステップS28に進み、制御部23は、AVデコーダ27を制御し、

デコード及び表示を停止させる。

次に、ステップS 2 9において、制御部2 3は、CM終了点のSTC_sequence_idに対応するSTC時間軸が開始するパケット番号を取得し、そのパケット番号と、ステップS 2 3の処理で取得したCM終了点のPTSとから、その点のPTSより時間的に前で、かつ、最も近いエントリポイントのあるソースパケット番号を取得する。

ステップS 3 0において、制御部2 3は、ステップS 2 9の処理で取得したエントリポイントのあるソースパケット番号から、トランスポートストリームのデータを読み出し、AVデコーダ2 7に供給させる。

ステップS 3 1において、制御部2 3は、AVデコーダ2 7を制御し、CM終了点のPTSのピクチャから表示を再開させる。

図1 0 7乃至図1 0 9を参照して、以上の動作をさらに説明すると、CM開始点とCM終了点は、この例の場合、STC_sequence_id=id0という共通のSTC時間軸上に存在し、そのSTC時間軸が開始するソースパケット番号は、シーンの開始点のソースパケット番号Aより小さいものとされている。

トランスポートストリームがデコードされ、ステップS 2 6で、表示時刻がPTS(b0)になったと判定された場合（CM開始点であると判定された場合）、AVデコーダ2 7により表示が停止される。そして、PTS(C)<PTS(c0)の場合、ステップS 3 0でパケット番号Cのデータから始まるストリームからデコードが再開され、ステップS 3 1において、PTS(c0)のピクチャから表示が再開される。

なお、この方法は、CMスキップ再生に限らず、一般的にClipMarkで指定される2点間のシーンをスキップして再生する場合にも、適用可能である。

次に、mark_entryとrepresentative_picture_entryが、図8 2に示すシンタクス構造である場合における、マーク点で示されるCMの頭出し再生処理について、図1 1 2のフローチャートを参照して説明する。

ステップS 4 1において、制御部2 3は、EP_map（図7 0）、STC_Info（図5 2）、Program_Info（図5 4）及びClipMark（図7 8）の情報を取得する。

次にステップS 4 2において、制御部2 3は、ステップS 4 1で読み出したClipMark（図7 8）に含まれるrepresentative_picture_entry（図8 2）又はref_thumbnail_indexで参照されるピクチャからサムネイルのリストを生成し、GUIの

メニュー画面上に表示させる。ref_thumbnail_indexが有効な値を有する場合、representative_picture_entryよりref_thumbnail_indexが優先される。

ステップS 4 3において、ユーザは再生開始点のマーク点を指定する。この指定は、例えば、ステップS 4 2の処理で表示されたメニュー画面上の中から、ユーザがサムネイル画像を選択し、そのサムネイルに対応付けられているマーク点を指定することで行われる。

ステップS 4 4において、制御部2 3は、ステップS 4 3の処理で指定されたマーク点のRSPN_ref_EP_startとoffset_num_pictures (図8 2) を取得する。

ステップS 4 5において、制御部2 3は、ステップS 4 4で取得したRSPN_ref_EP_startに対応するソースパケット番号からトランスポートストリームのデータを読み出し、AVデコーダ2 7に供給させる。

ステップS 4 6において、制御部2 3は、AVデコーダ2 7を制御し、RSPN_ref_EP_startで参照されるピクチャから（表示はしないで）、表示すべきピクチャをカウントアップしていき、カウント値がoffset_num_picturesになったとき、そのピクチャから表示を開始させる。

以上の処理を、図1 1 3乃至図1 1 5を参照して、さらに説明する。この例においては、DVRトランスポートストリームファイルは、ソースパケット番号Aからシーンが開始しており、ソースパケット番号BからソースパケットCまでCMが挿入されている。このため、図1 1 4に示されるように、EP_mapには、RSPN_EP_startとしてのA、B、Cに対応して、PTS_EP_startとして、PTS(A)、PTS(B)、PTS(C)が登録されている。

また、図1 1 5に示されるように、シーンスタート、CMスタート及びCMエンドのマークタイプに対応して、mark_entryとrepresentative_picture_entryが登録されている。mark_entryには、シーンスタート、CMスタート及びCMエンドに対応して、RSPN_ref_EP_startとして、それぞれA、B、Cが登録され、offset_num_picturesとして、M 1、N 1、N 2 が登録されている。同様に、representative_picture_entryには、RSPN_ref_EP_startとして、シーンスタート、CMスタート及びCMエンドに対応して、それぞれA、B、Cが登録され、offset_num_picturesとして、M 2、N 1、N 2 がそれぞれ登録されている。

シーンスタートに当たるピクチャから頭出して再生が指令された場合、パケット番号Aのデータから始まるストリームからデコードが開始され、PTS(A)のピクチャから（表示をしないで）表示すべきピクチャをカウントアップをしていき、offset_num_picturesが、M1の値になったとき、そのピクチャから表示が開始される。

さらに、mark_entryとrepresentative_picture_entryのシンタクスが、図82に示される構成である場合におけるCMスキップ再生の処理について、図116のフローチャートを参照して説明する。

ステップS61において、制御部23は、EP_map（図70）、STC_Info（図52）、Program_Info（図54）及びClipMark（図78）の情報を取得する。

ステップS62において、ユーザがCMスキップ再生を指令すると、ステップS63において、制御部23は、マークタイプ（図79）がCM開始点とCM終了点である各点のマーク情報として、RSPN_ref_EP_startとoffset_num_pictures（図82）を取得する。そして、CM開始点のデータは、RSPN_ref_EP_start(1)、offset_num_pictures(1)とされ、CM終了点のデータは、RSPN_ref_EP_start(2)、offset_num_pictures(2)とされる。

ステップS64において、制御部23は、RSPN_ref_EP_start(1)、RSPN_ref_EP_start(2)に対応するPTSをEP_map（図70）から取得する。

ステップS65において、制御部23は、トランスポートストリームを記録媒体100から読み出させ、AVデコーダ27に供給させる。

ステップS66において、制御部23は、現在の表示画像がRSPN_ref_EP_start(1)に対応するPTSのピクチャであるか否かを判定し、現在の表示画像がRSPN_ref_EP_start(1)に対応するPTSのピクチャでない場合には、ステップS67に進み、ピクチャをそのまま継続的に表示させる。その後、処理はステップS65に戻り、それ以降の処理が繰り返し実行される。

ステップS66において、現在の表示画像がRSPN_ref_EP_start(1)に対応するPTSのピクチャであると判定された場合、ステップS68に進み、制御部23は、AVデコーダ27を制御し、RSPN_ref_EP_start(1)に対応するPTSのピクチャから表示するピクチャをカウントアップしていき、カウント値がoffset_num_pictures

(1)になったとき、表示を停止させる。

ステップS 6 9において、制御部2 3は、RSPN_ref_EP_start(2)のソースパケット番号からトランスポートストリームのデータを読み出し、AVデコーダ2 7に供給させる。

ステップS 7 0において、制御部2 3は、AVデコーダ2 7を制御し、RSPN_ref_EP_start(2)に対応するPTSのピクチャから（表示をしないで）表示すべきピクチャをカウントアップしていき、カウント値がoffset_num_pictures(2)になったとき、そのピクチャから表示を開始させる。

以上の動作を、図1 1 3乃至図1 1 5を参照してさらに説明すると、まず、EP_map（図1 1 4）をもとに、パケット番号B、Cに対応する時刻PTS(B)、PTS(C)が得られる。そして、Clip AV streamがデコードされていき、表示時刻がPTS(B)になったとき、PTS(B)のピクチャから表示ピクチャがカウントアップされ、その値がN 1（図1 1 5）になったとき、表示が停止される。

さらに、パケット番号Cのデータから始まるストリームからデコードが再開され、PTS(C)のピクチャから（表示をしないで）表示すべきピクチャをカウントアップしていき、その値がN 2（図1 1 5）になったとき、そのピクチャから表示が再開される。

以上の処理は、CMスキップ再生に限らず、ClipMarkで指定された2点間のシーンをスキップさせて再生する場合にも、適用可能である。

次に、mark_entryとrepresentative_picture_entryのシンタクスが、図8 4に示すような構成である場合における、マーク点で示されるシーンの頭出し再生処理について、図1 1 8のフローチャートを参照して説明する。

ステップS 8 1において、EP_map（図7 0）、STC_Info（図5 2）、Program_Info（図5 4）、並びにClipMark（図7 8）の情報が取得される。

ステップS 8 2において、制御部2 3は、ClipMark（図7 8）のrepresentative_picture_entry又はref_thumbnail_indexで参照されるピクチャからサムネイルのリストを生成し、GUIのメニュー画面として表示させる。ref_thumbnail_indexが有効な値を有する場合、representative_picture_entryよりref_thumbnail_indexが優先される。

ステップS 8 3において、ユーザは再生開始点のマーク点を指定する。この指定は、例えば、メニュー画面上の中からユーザがサムネイル画像を選択し、そのサムネイルに対応付けられているマーク点を指定することで行われる。

ステップS 8 4において、制御部23は、ユーザから指定されたmark_entryのRSPN_mark_point(図84)を取得する。

ステップS 8 5において、制御部23は、マーク点のRSPN_mark_pointより前にあり、かつ、最も近いエントリポイントのソースパケット番号を、EP_map(図70)から取得する。

ステップS 8 6において、制御部23は、ステップS 8 5で取得したエントリポイントに対応するソースパケット番号からトランスポートストリームのデータを読み出し、AVデコーダ27に供給させる。

ステップS 8 7において、制御部23は、AVデコーダ27を制御し、RSPN_mark_pointで参照されるピクチャから表示を開始させる。

以上の処理を、図119乃至図121を参照してさらに説明する。この例においては、DVRトランスポートストリームファイルが、ソースパケットAでシーンスタートし、ソースパケット番号BからCまでCMが挿入されている。このため、図120のEP_mapには、RSPN_EP_startとしてのA、B、Cに対応して、PTS_EP_startがそれぞれPTS(A)、PTS(B)、PTS(C)として登録されている。また、図121に示されるClipMarkに、シーンスタート、CMスタート及びCMエンドに対応して、markentryのRSPN_mark_pointとして、a1、b1、c1が、また、representative_picture_entryのRSPN_mark_pointとして、a2、b1、c1が、それぞれ登録されている。

シーンスタートにあたるピクチャから頭出して再生する場合、パケット番号A < a1とすると、パケット番号Aのデータから始まるストリームからデコードが開始され、ソースパケット番号a1に対応するピクチャから表示が開始される。

次に、mark_entryとrepresentative_picture_entryのシンタクスが、図84に示されるような構成である場合におけるCMスキップ再生の処理について、図122と図123のフローチャートを参照して説明する。

ステップS 101において、制御部23は、EP_map(図70)、STC_Info(図

52)、Program_Info (図54)、並びにClipMark (図70) の情報を取得する。

ステップS102において、ユーザは、CMスキップ再生を指定する。

ステップS103において、制御部23は、マークタイプ (図79) がCM開始点とCM終了点である各点のマーク情報のRSPN_mark_point (図84) を取得する。そして、制御部23は、CM開始点のデータをRSPN_mark_point (1) とし、CM終了点のデータをRSPN_mark_point (2) とする。

ステップS104において、制御部23は、記録媒体100からトランスポートストリームを読み出させ、AVデコーダ27に出力し、デコードさせる。

ステップS105において、制御部23は、現在の表示画像がRSPN_mark_point (1) に対応するピクチャであるか否かを判定し、現在の表示画像がRSPN_mark_point (1) に対応するピクチャでない場合には、ステップS106に進み、そのままピクチャを継続的に表示させる。その後、処理はステップS104に戻り、それ以降の処理が繰り返し実行される。

ステップS105において、現在の表示画像がRSPN_mark_point (1) に対応するピクチャであると判定された場合、ステップS107に進み、制御部23はAVデコーダ27を制御し、デコード及び表示を停止させる。

次に、ステップS108において、RSPN_mark_point (2) より前にあり、かつ、最も近いエントリポイントのあるソースパケット番号がEP_map (図70) から取得される。

ステップS109において、制御部23は、ステップS108で取得したエントリポイントに対応するソースパケット番号からトランスポートストリームのデータを読み出し、AVデコーダ27に供給させる。

ステップS110において、制御部23は、AVデコーダ27を制御し、RSPN_mark_point (2) で参照されるピクチャから表示を再開させる。

以上の処理を図119乃至図121の例でさらに説明すると、Clip AV stream をデコードしていき、ソースパケット番号b1 (図121) に対応する表示ピクチャになったとき、表示が停止される。そして、ソースパケット番号C<ソースパケット番号c1とすると、パケット番号Cのデータから始まるストリームからデコードが再開され、ソースパケット番号c1に対応するピクチャになったとき、

そのピクチャから表示が再開される。

以上のようにして、図 1 2 4 に示されるように、PlayList上で、タイムスタンプにより所定の位置を指定し、このタイムスタンプを各ClipのClip Informationにおいて、データアドレスに変換し、Clip AV streamの所定の位置にアクセスすることができる。

より具体的には、図 1 2 5 に示されるように、PlayList上において、PlayList Markとしてブックマークやリジューム点を、ユーザが時間軸上のタイムスタンプとして指定すると、そのPlayListは再生するとき、そのPlayListが参照しているClipのClipMarkを使用して、Clip AV streamのシーン開始点やシーン終了点にアクセスすることができる。

なお、ClipMarkのシンタクスは、図 7 8 の例に代えて、図 1 2 6 に示すようにすることもできる。

この例においては、RSPN_markが、図 7 8 のreserved_for_MakerID、mark_entry () 及びrepresentative_picture_entry () に替えて挿入されている。このRSPN_markの 3 2 ビットのフィールドは、AVストリームファイル上で、そのマークが参照するアクセスユニットの第 1 バイト目を含むソースパケットの相対アドレスを示す。RSPN_markは、ソースパケット番号を単位とする大きさであり、AVストリームファイルの最初のソースパケットからClip Information fileにおいて定義され、offset_SPNの値を初期値としてカウントされる。

その他の構成は、図 7 8 における場合と同様である。

ClipMarkのシンタクスは、さらに図 1 2 7 に示すように構成することもできる。この例においては、図 1 2 6 におけるRSPN_markの代わりに、RSPN_ref_EP_startとoffset_num_picturesが挿入されている。これらは、図 8 2 に示した場合と同様のものである。

次に、図 1 2 8 は、ClipInfo()のシンタクスの別例を示す。

Clip_service_typeは、AVストリームファイルのタイプを示す。例えば、Clip_service_typeは、ビデオレコーディングやオーディオレコーディングなどのタイプを示す。また、例えば、Clip_service_typeは、デジタルTV放送のプログラムが示すサービスタイプと同じ意味を持たせてもよい。例えば、日本のデジタルBS

放送の場合、サービスタイプは、テレビジョンサービス、音声サービス及びデータ放送サービスの3種類を持つ。AVストリームが含むプログラムのサービスタイプを代表する値をClip_service_typeにセットする。

transcode_mode_flagは、デジタル放送から受信されたMPEG-2トランスポートストリームの記録方法を示すフラグである。このフラグが1にセットされている場合、Clipに対応するAVストリームファイル中の少なくとも1つのエレメンタリストリームは再符号化されて記録されたことを示す。このフラグが1にセットされている場合、Clipに対応するAVストリームファイル中の全てのエレメンタリストリームはデジタル放送から受信されたままの内容で何も変更されないで記録されたことを示す。

その他のシンタクスフィールドは、図46で説明した同名のフィールドと同じ意味を持つ。

次に、図129を参照して、ProgramInfo()の別例について説明する。

AVストリームファイルの中で本フォーマットが規定するところのプログラム内容が一定であるソースパケット列を、program-sequenceと呼ぶ。

AVストリームファイルの中で、新しいprogram-sequenceが開始するアドレスをProgramInfo()にストアする。このアドレスは、SPN_program_sequence_startにより示される。

AVストリームファイルの中にある最後のprogram-sequence以外のprogram-sequenceは、そのSPN_program_sequence_startで指されるソースパケットから開始し、その次のSPN_program_sequence_startで指されるソースパケットの直前のソースパケットで終了する。最後のprogram-sequenceは、そのSPN_program_sequence_startで指されるソースパケットから開始し、AVストリームファイルの最後のソースパケットで終了する。

program-sequenceは、STC-sequenceの境界をまたいてもよい。

lengthは、このlengthフィールドの直後のバイトからProgramInfo()の最後のバイトまでのバイト数を示す。

num_of_program_sequencesは、AVストリームファイルの中にあるprogram-sequenceの数を示す。

SPN_program_sequence_startは、AVストリームファイル上でprogram-sequenceが開始するアドレスを示す。SPN_program_sequence_startは、ソースパケット番号を単位とする大きさであり、AVストリームファイルの最初のソースパケットから、0を初期値としてカウントされる。

ProgramInfo()の中でエントリされるSPN_program_sequence_startの値は、昇順に並んでいる。

SPN_program_sequence_startは、そのprogram_sequenceに対する最初のPMTを持つソースパケットを指していることを前提とする。SPN_program_sequence_startは、記録機（図1の記録再生装置1）がトランスポートストリーム中のPSI/SIを解析することによって作られる。記録機がPSI/SIを解析し、その変化を検出するまでの遅延時間が必要なために、SPN_program_sequence_startは、実際のPSI/SIの変化点から所定の時間以内にあるソースパケットを指してもよい。

program_map_PIDは、そのprogram-sequenceに適用できるPMT(program map table)を持つトランスポートパケットのPIDの値である。

num_of_streams_in_psは、そのprogram-sequenceの中で定義されるエレメンタリストリームの数を示す。

num_of_groupsは、そのprogram-sequenceの中で定義されるエレメンタリストリームのグループの数を示す。num_of_groupsは、1以上の値である。

トランスポートストリームのPSI/SIがエレメンタリストリームのグループ情報を持つ場合、num_of_groupsは、1以上の値をとることを想定している。それぞれのグループは、マルチビュープログラム中の1つのビューを構成する。

stream_PIDは、そのprogram-sequenceのprogram_map_PIDが参照するところのPMTの中で定義されているエレメンタリストリームに対するPIDの値を示す。

StreamCodingInfo()は、上記stream_PIDで指されるエレメンタリストリームの情報を示す。詳細は後述する。

num_of_streams_in_groupは、エレメンタリストリームのグループが持つエレメンタリストリームの数を示す。

stream_indexは、上記エレメンタリストリームのグループが持つエレメンタリストリームに対応するところの、シンタクス中のstream_indexのfor-loopで記述